



HAL
open science

From data exploration to presentation: designing new systems and interaction techniques to enhance the sense-making process

Hugo Romat

► **To cite this version:**

Hugo Romat. From data exploration to presentation: designing new systems and interaction techniques to enhance the sense-making process. Human-Computer Interaction [cs.HC]. Université Paris Saclay (COMUE), 2019. English. NNT: 2019SACLS335 . tel-02390645

HAL Id: tel-02390645

<https://theses.hal.science/tel-02390645>

Submitted on 3 Dec 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

From data exploration to presentation: designing new systems and interaction techniques to enhance the sense-making process

Thèse de doctorat de l'Université Paris-Saclay
préparée à l'Université Paris Sud

Ecole doctorale n°580 Sciences et Technologies de l'Information et de la
Communication (STIC)
Spécialité de doctorat : Informatique

Thèse présentée et soutenue à Orsay, le 3 Octobre 2019, par

HUGO ROMAT

Composition du Jury :

Stéphane Huot Directeur de Recherche, Inria Lille	Président
Daniel Wigdor Professeur, University of Toronto	Rapporteur
David Auber Professeur, Université de Bordeaux	Rapporteur
Uta Hinrichs Professeur, University of St Andrews	Examineur
Jean-Daniel Fekete Directeur de Recherche, Inria Saclay	Examineur
Caroline Appert Directeur de Recherche, CNRS	Co-directeur de thèse
Emmanuel Pietriga Directeur de Recherche, Inria Saclay	Directeur de thèse
Thomas Honoré Chef de projet, TecKnowMetrix	Invité
Christophe Lecante CEO, TecKnowMetrix	Invité

Abstract

Au cours de la dernière décennie, la quantité de données n'a cessé d'augmenter. Ces données peuvent provenir de sources variées, telles que des smartphones, des enregistreurs audio, des caméras, des capteurs, des simulations, et peuvent avoir différentes structures. Bien que les ordinateurs puissent nous aider à traiter ces données, c'est le jugement et l'expertise humaine qui les transforment réellement en connaissances. Cependant, pour donner un sens à ces données de plus en plus diversifiées, des techniques de visualisation et d'interaction sont nécessaires. Ce travail de thèse contribue de telles techniques pour faciliter l'exploration et la présentation des données, lors d'activités visant à faire sens des données.

Dans la première partie de cette thèse, nous nous concentrons sur les systèmes interactifs et les techniques d'interaction pour aider les utilisateurs à faire sens des données. Nous étudions comment les utilisateurs travaillent avec des contenus divers afin de leur permettre d'externaliser leurs pensées par le biais d'annotations digitales. Nous présentons notre approche avec deux systèmes. Le premier, ActiveInk, permet l'utilisation naturelle du stylet pour la lecture active, lors d'un processus d'exploration de données. Dans le cadre d'une étude qualitative menée auprès de huit participants, nous contribuons des observations sur les comportements de la lecture active au cours de l'exploration des données, et, des principes aidant les utilisateurs à faire sens des données. Le second système, SpaceInk, est un espace de conception de techniques en utilisant le stylet et les gestes, qui permet de créer de l'espace pour les annotations, pendant la lecture active, en ajustant dynamiquement le contenu du document.

Dans la deuxième partie de cette thèse, nous avons étudié les techniques permettant de représenter visuellement les éléments de réponses aux questions quand les utilisateurs essaient de faire sens des données. Nous nous concentrons sur l'une des structures de données les plus élaborées : les réseaux multi-variés, que nous visualisons à l'aide de diagrammes noeuds-liens. Nous étudions comment permettre un processus de conception itératif flexible lors de la création de diagrammes noeuds-liens pour les réseaux multi-variés. Nous présentons d'abord un système, Graphies, qui permet la création de visualisations expressives de diagrammes noeuds-liens en fournissant aux concepteurs un environnement de travail flexible qui rationalise le processus créatif et offre un support efficace pour les itérations rapides de conception. Allant au-delà de l'utilisation de variables visuelles statiques dans les diagrammes noeuds-liens, nous avons étudié le potentiel des variables liées au mouvement pour encoder les attributs des données.

En conclusion, nous montrons dans cette thèse que le processus visant à faire sens des données peut être amélioré à la fois dans le processus d'exploration et de présentation, en utilisant l'annotation comme nouveau moyen de transition entre exploration et externalisation, et en suivant un processus itératif et flexible pour créer des représentations expressives de données. Les systèmes qui en résultent établissent un cadre de recherche où la présentation et l'exploration sont au cœur des systèmes de données visuelles.

Acknowledgements

I would like to thank everybody who supported and accompanied me in the whole journey of my Ph.D.

First I would like to thank my two advisors, for allowing me to undertake this Ph.D. about a topic I am truly passionate, introducing me to HCI research, having always believed in me, and after all for teaching me so much. Thank you Caroline for the encouragement and for always being available when I needed. Thank you Emmanuel for all the support during this thesis, for teaching me what rigor is, and, of course for teaching me about the right way to use colors.

Thank you also Christophe, Thomas, and all the members of TecKnowMetrix for the support along this thesis, and the insightful discussions we had. Thank you also Nathalie and Ken, and other members of the EPIC team for the opportunities you gave me, for having confidence in my capabilities and for inspiring me.

Then, I would like to thanks all the members of the ILDA team. Thanks to Anastasia and Olivier for advice and help you gave me when I needed them. Thanks to Arnaud for all the highly scientific conversation we had, at the lab, and abroad. Thanks Maria, for your patience, and for you support. Thanks to Eugenie for all the discussions, and breaks we have shared. And thanks to all the other students, post-docs and engineers for the good atmosphere in the team. Thanks you also Marc, Bruno and Manos for all the discussion we had about research and all the moments we have shared together.

Outside of research, I would like to thanks my parents and my sister; Chloe. It was a difficult time, but I finally did it, thanks to you. Thanks to my bunch of childhood friends; Jason, Jess, Louis, Bob, Bastien, Anne-So, Leo and all the others, to be here for me, and for all the support you gave me since so many years. Thank you Cailloux for all those beautiful adventures we have been through. Thank you Romain and Quentin, for beeing such friends and for all those memorable nights we spent, and we will spend together. Thank you Dede, Gaetan, Stephanie for all the moments we have shared. Thank you Lea, Guillaume, Jessie and Zelie, for all marine adventures we went through and for such a beautiful friendship.

And thanks Pauline to be always here for me.

Contents

Abstract	2
Introduction	10
0.1 Research Question	12
0.2 Thesis Overview	12
I Making sense of varied types of data using ink as a medium	15
1 Using ink to transition between interacting with data and externalizing thoughts	17
1.1 Related Work	19
1.1.1 Pen and Touch Visualization Systems	19
1.1.2 Unique Affordances of Digital Pens	19
1.1.3 Externalization in Visualization	20
1.2 ActiveInk	20
1.2.1 Design Principles	21
1.2.2 Unified Infinite Canvas	22
1.2.3 Pen Actions	23
1.2.4 Prefix Interface: Switching between Types of Pen	24
1.2.5 Postfix Interface: Activating Ink Strokes	24
1.2.6 Implementation	25
1.3 Qualitative Study	25
1.3.1 Hardware Capabilities	25
1.3.2 Research questions	25
1.3.3 Study Design	26
1.3.3.1 Ink	26
1.3.3.2 ActiveInk Prefix and Postfix	26
1.3.4 Participants and Method	26
1.3.4.1 Data and Task	27
1.3.4.2 Procedure	27
1.3.4.3 Data Collection & Data Analysis	27
1.3.5 Results	28
1.3.5.1 Ink	28
1.3.5.2 Externalizations with ActiveInk	29
1.3.5.3 Sensemaking strategies with ActiveInk	31
1.3.5.4 Ratings and Preferences	31

1.3.5.5	Wishes	32
1.3.6	Summary and Discussion	32
1.3.7	Implications for Design	33
1.3.8	Advanced Actions to Support Sensemaking	33
1.4	Conclusion and Future Work	34
2	Making space for externalizing thoughts on reflowable documents that support various content	37
2.1	Related Work	39
2.1.1	Digital Active Reading	39
2.1.2	Active Annotations	39
2.1.3	Overlay Ink and Reflowable Documents	40
2.2	SpaceInk	41
2.2.1	Design Space	41
2.2.2	Implemented Techniques	44
2.3	User Study	46
2.3.1	Task	47
2.3.2	Participants and Apparatus	47
2.3.3	Procedure	47
2.3.3.1	Data Collection & Data Analysis	48
2.3.4	Results	48
2.3.4.1	People use the full range of strategies	48
2.3.4.2	People do not estimate required space accurately	50
2.3.4.3	Less agency may prove more comfortable	51
2.4	In-Context Annotations In Practice	52
2.4.1	Seamless Transitions between Inks (R_1)	52
2.4.2	Combining Strategies for Making Space (R_2)	55
2.4.3	Aligning Ink(s) and Content (R_3)	57
2.5	Future Work	57
II	Presenting results of sense-making activities through an expressive design approach	59
3	Creating expressive node-link diagrams using Graphies	61
3.1	Related Work	62
3.1.1	General-purpose Expressive Visualization Design	62
3.1.2	Authoring Graph Visualizations	63
3.1.3	Visual Exploration of Multivariate Graphs	64
3.2	Design process	65
3.3	Graphies	67
3.3.1	Populating the Canvas with Data	68
3.3.2	Customizing the Layout	71
3.3.3	Creating Visual Mappings	71
3.3.4	Supporting an Iterative Design Process	73
3.3.5	Exporting Visual Designs	73
3.3.6	Implementation	74
3.4	User study	74
3.4.1	Procedure	74
3.4.2	Participants & Apparatus	75

CONTENTS

3.4.2.1	Data Collection	75
3.4.3	Results	76
3.5	Discussion, Feedback and Future Work	76
3.5.1	Observations	78
3.5.2	Feedback from Study Participants	78
3.5.3	Feedback from Data Analysts	79
4	Using motion as an encoding channel in node-link diagrams	81
4.1	Background and Motivation	82
4.1.1	Motion in Human-Computer Interaction	83
4.1.2	Motion Perception	84
4.2	Animated Edge Textures	85
4.2.1	Model	85
4.2.2	API	86
4.2.3	Prototype Implementation	87
4.3	Examples of Use	87
4.3.1	Encoding a single attribute	88
4.3.1.1	Encoding a numerical attribute.	88
4.3.1.2	Reducing Visual Complexity	88
4.3.1.3	Encoding a categorical attribute.	90
4.3.2	Encoding multiple attributes	90
4.4	Experiment 1: Encoding edge attributes with motion variables only	91
4.4.1	Difference Threshold for <i>Speed</i> and <i>Frequency</i>	92
4.4.2	Encoding Edge Attributes	92
4.4.2.1	Task and Procedure	92
4.4.2.2	Participants & Apparatus	94
4.4.2.3	Data Collection & Data Analysis	96
4.4.2.4	Results	96
4.4.3	Follow-Up Study	97
4.4.3.1	Participants & Apparatus	98
4.4.3.2	Task & Procedure	98
4.4.3.3	Results	98
4.5	Experiment 2: Encoding edge attributes with both static and motion variables	99
4.5.1	Task	100
4.5.2	Participants & Apparatus	104
4.5.3	Results	104
4.5.4	Summary of findings	108
4.6	Conclusion and Future Work	109
	Conclusion	110
4.6.1	Summary	112
4.6.2	Perspectives	113
4.6.2.1	Optimizing interaction techniques based on hardware capabilities	113
4.6.2.2	Exploration vs Presentation	114
	Bibliography	117

A	Appendix	135
A.1	Detailed instructions participants received during Graphies' study .	136
A.2	Canvas created by participants during ActiveInk study	141

List of publications

- [1] Hugo Romat, Caroline Appert, Benjamin Bach, Nathalie Henry-Riche, and Emmanuel Pietriga. Animated edge textures in node-link diagrams: A design space and initial evaluation. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 187:1–187:13, New York, NY, USA, 2018. ACM.
- [2] Hugo Romat, Nathalie Henry Riche, Ken Hinckley, Bongshin Lee, Caroline Appert, Emmanuel Pietriga, and Christopher Collins. Activeink: (th)inking with data. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, pages 42:1–42:13, New York, NY, USA, 2019. ACM.
- [3] Hugo Romat, Dylan Lebout, Emmanuel Pietriga, and Caroline Appert. Animated particles in node-link diagrams: Influence of particles' visual appearance on their perceived speed. In *INTERACT'19: IFIP International Conference on Human-Computer Interaction*, Cyprus, Croatia, September 2019. Springer.
- [4] Hugo Romat, Emmanuel Pietriga, Nathalie Henry Riche, Ken Hinckley, and Caroline Appert. Spaceink: Making space for in-context annotations. In *Proceedings of the 32th Annual ACM Symposium on User Interface Software and Technology*, UIST '19, pages 107–112, New York, NY, USA, 2019. ACM.

Introduction

The rate at which data gets produced keeps increasing: user-generated content disseminated through social networks; new Web sites (one is created every minute on average); scientific and technical data; business-related data; open government data and data published by public institutions such as national libraries. Taken together, these represent a spectacular wealth of very diverse information. However, value can be gained from these data only if we are able to make sense of them and find the relevant pieces of information in this overwhelming **volume** of data.

Making sense of data can mean different things, ranging from understanding a simple table in a document to surveying the scientific and technical literature on a given topic to get a clear picture of opportunities for technological innovation in that area. In the latter case, subject-matter experts have to extract and cross-reference information from multiple sources, often relying on visual representations of the data to help them in their sense-making activity. The **variety** in data sources often adds value given the complementary nature of the data held by each source, but also means that multiple visual representations are required to convey those data to users, that provide relevant, understandable views on the data, supporting tasks such as identifying trends, locating elements in their spatial-temporal context, *etc.* Only then can users turn the data collection into actual knowledge, using those representations as support for understanding as well as reasoning: biologists looking at the interaction between proteins, data scientists investigating the impact of a patent in the scientific literature, or sociologists looking at relationships between people.

Beyond the volume of data and the variety of sources, another challenge knowledge workers have to deal with is the multiple ways we can give **structure** to data: raw text, quantitative data stored in simple tables, relation databases, structured document formats, semi-structured data distributed over the Web, multivariate networks, *etc.* Some have a clearly defined model, strongly enforced by the schema that describes them, which in turn makes exploration of the data themselves easier, if only by defining more clearly the perimeter of possible queries. Other formats provide more flexibility, at the cost of heterogeneity, irregularity and incompleteness. Again, users are confronted to the difficulty of having to deal with different formats and different data structures in their sense-making activities, as they combine data from multiple sources.

Significant progress has been made over the years in our ability to collect, store and query large volumes of varied data. However, there are still strong bottlenecks in sense-making activities, one of them being our difficulty to quickly find relevant information in large, heterogeneous dataset collections and represent them in a meaningful manner. This PhD thesis is about helping users in various activities that relate to manipulating, and making sense of, data thanks to interactive systems, with a focus on two types of such systems:

- domain-specific tools focused on one specific type of data, used by subject-matter experts;
- more general-purpose tools that target a broader range of knowledge workers.

Domain-specific Tools for Subject-matter Experts

Work on this CIFRE PhD thesis has been conducted at Inria and in collaboration with a French SME: TecKnowMetrix (TKM), in which I was hired as a member of the Research and Development department. TKM is a consulting company that develops

methods and tools for analyzing scientific and technological information in a given technological domain. Applying these methods to large volumes of data (patents, scientific publications, *etc.*) enables TKM analysts to produce a synthetic vision of a complex domain. The job TKM analysts are tasked with consists of gathering the relevant scientific literature, exploring and analyzing it in order to gain insights about this domain and answer specific questions TKM customers have about it. Visualizing those large volumes of data and document collections is essential, as it helps analysts not only understand and analyze the data, but also communicate findings about it to customers.

During my PhD, I had the opportunity to follow 4 TKM analysts during 3 years. I observed the main steps of the various activities performed by these experts in the workflow of a typical project: (i) *Data collection/Creation*: Analysts first survey the literature to find documents related to a project question such as, *e.g.*, identifying emerging technologies in a particular domain. Such a survey will yield several thousands of patents and publications, that they store in a database. (ii) *Visualizing, Exploring and Analyzing*: Once all documents and data have been captured and stored in that relational database with the corresponding descriptive metadata, they explore the collection by visualizing its items based on those metadata. (iii) *Presenting*: The resulting insights and answers to specific questions are compiled into a report, often using graph-based visual representations that capture the dependencies and other relationships between items.

I focused part of my PhD work on that latter step: investigating ways to support users in the authoring of visual representations of complex, multivariate networks. Indeed, the current authoring workflow for producing such visual representations is suboptimal. Analysts use network visualizations tools to produce an initial representation of their graph, but then almost systematically switch to a general-purpose vector-graphics editor to fine-tune the visual appearance of the resulting node-link diagram, using graphics editing capabilities that are not available in the first family of tools. This makes the process tedious and incompatible with an iterative graphical design approach, even more so when considering that any operation that requires regenerating the diagram from the actual multivariate network data requires going back to the network visualization tool and then performing all customizations again.

General-purpose Tools for a Broad Range of Knowledge Workers

The target audience of tools such as those described above is relatively limited, even though it goes well beyond TKM analysts and actually encompasses all authors of node-link diagrams. In the context of a 3-month internship in the EPIC team at Microsoft Research in Redmond during the fall of 2018, I had the opportunity to investigate another context of use still related to data sense-making, but this time targeting the broader audience of knowledge workers at large, who use software suites such as Microsoft Office. Such software suites contain multiple applications, including a word processor (*e.g.*, MS-Word or Apple Pages), a spreadsheet editor (*e.g.*, MS-Excel or Apple Numbers), a presentation application (*e.g.*, MS-Powerpoint or Apple Keynote). These tools are easy-to-use and generic enough to be used by a wide variety of users, ranging from business analysts to scientists from all disciplines, computer scientists, researchers, office workers and team managers.

Each of these applications specializes in the interactive editing and viewing of essentially one type of data: text documents, tabular data, *etc.*. But even though

they are all grouped in *software suites*, it remains difficult to merge those different data in a single workspace. For instance, a word processor will often have some limited support for tables, but that remains far from the capabilities of a spreadsheet editor. Consequently, users need to switch between applications both to manipulate the data and to analyze them.

Recently, Microsoft has started moving toward implementing a vision based on a single workspace by making data of various types coexist in a single application: OneNote. In this application, users interact with content laid out on an infinite canvas. They can bring different types of content and manipulate them. Common actions are supported such as copying, pasting, resizing and moving items. However, the interaction is limited to a common denominator, making this application not particularly more powerful than a paper notebook. This severely limits the efficiency of such tools, and even if such systems could support both the viewing and in-situ editing of a wide range of data types and document formats, the way users interact with these varied content types would still have to be deeply reconsidered.

0.1 Research Question

In both contexts of use (general-purpose tools for a broad range of knowledge workers & domain-specific tools for subject-matter experts), users are likely to be confronted to significant complexity in the data they have to handle.

In the former case, one challenge is to enable people to more actively engage and think with data. General-purpose tools should better adapt interaction to the diversity of content, as well as provide a better integration between operations aimed at manipulating content and operations aimed at annotating that content, thereby externalizing thoughts. The first part of this PhD thesis makes contributions in this direction, that can be summed up with the following research question:

*How to enhance sense-making activities
when working with multiple types of content?*

In the latter case, one challenge is to enable people to author more expressive visualizations of complex data structures such as multivariate networks, supporting a more flexible and iterative design process. Interactive authoring environments should provide a flexible workflow that streamlines the creative process and effectively supports quick design iterations. They should support complex data structures and elaborate visual mappings. Support for the exploration of several design alternatives is also key to effectively enabling an iterative design workflow. The second part of this PhD thesis makes contributions in this direction, that can be summed up with the following research question:

*How to make the authoring of expressive visualizations
for complex data structures more efficient?*

0.2 Thesis Overview

During sense-making and active-reading activities, people annotate documents and data with their insights: they underline sentences in a document, circle regions on

a map or underline a part of a text document. They jot down their hypotheses in the margin, draw correlation lines on scatterplots, or create personal legends to track patterns. Annotations are central to the active-reading and sense-making process. In the first part of this thesis, I focus on *ink* as a medium to seamlessly transition between interacting with data and externalizing thoughts using a digital pen. This general idea is at the core of the first two chapters, that describe an interactive environment for sense-making from varied data sources, and interaction techniques for annotating documents, respectively. Each chapter describes the general approach, the software prototype implementation, and the studies that have been conducted to gather feedback and validate the approach.

In Chapter 1, I introduce ActiveInk, a system that enables people to actively engage and think with data. ActiveInk enables users to manipulate visual representations of data from varied sources laid out on an infinite canvas, as well as annotate them. Each element on the canvas can be manipulated and rearranged. Instead of keeping users' annotations passive, the system lets users turn them into actions. It considers each annotation as a deferred command that can be activated later. Pen+touch interaction enables users to take notes as well as manipulate data.

When editing or reviewing a document, people directly overlay ink marks on content. For instance, they underline words, or circle elements in a figure. In Chapter 2, I introduce SpaceInk, a design space of pen+touch techniques that make room (white space) for placing annotations in text documents by dynamically reflowing a document's content. SpaceInk lets users concentrate on capturing fleeting thoughts, streamlining the overall annotation process by enabling the fluid interleaving of space-making gestures with freeform ink. SpaceInk focuses specifically on text documents, leveraging the power of digital inking for active reading.

While the first two chapters focus on interactive systems and techniques as a means to support sense-making activities, the second part of this thesis focuses on means to visually present the results of such activities: questions answered and insights gained from data analysis. Authors of such presentations will often seek to convey a specific message through data visualization. I focus here on an intrinsically complex type of data: multivariate networks. Again, each chapter describes the general approach, the software prototype implementation, and the studies that have been conducted to gather feedback and validate the approach.

In Chapter 3 I introduce Graphies, an environment for authoring network visualizations for communication purpose. Informed by a user-centered design process involving TKM analysts, Graphies has been developed as an expressive design environment for node-link diagrams supporting a flexible and iterative design process. Graphies lets designers incrementally populate an infinite canvas with nodes and links of interest from multivariate graph datasets; specify visual mappings and apply them to arbitrary selections of such elements; and customize the visual appearance of individual elements as well.

Graphies enables users to create elaborate visual mappings that can involve a wide range of visual variables. This set of variables, however, remains limited in classic node-link diagrams, especially when considering the available variables for edges. In Chapter 4, I study the potential of motion variables to encode additional information on edges in node-link diagrams. Our model relies on particles flowing along the links, represented as animated edge textures. I describe the model and associated Web-based

CONTENTS

framework for generating animated edge textures, and illustrate its capabilities using different examples of visual mappings. I report on two evaluations of particle properties in terms of visual perception. An initial evaluation that studies the perception of each motion variable in isolation, and a second evaluation that evaluates the influence of two visual variables, particle color and particle size, on the perception of particle speed across links in a diagram.

PART I

Making sense of varied types of data using ink as a medium

Analysts often get insights by bringing complementary, heterogeneous data and documents, comparing and cross referencing them. Digital annotations help these analysts make sense of the information coming from these different sources, as they enable them to externalize their ideas, jotting down comments directly on those documents. Annotations can also serve other purposes such as emphasizing important parts of the information, creating reminders for later reading, or sketching actions to be performed on the data. However, working with such diversity of content, and being able to externalize thoughts through digital annotations, are difficult tasks that current interactive systems do not support well, especially in the case of subject-matter experts such as data analysts who perform advanced sense making tasks.

The two chapters in this first part describe work that aimed at better supporting the sense-making process using digital ink to transition between externalization and exploration; and interaction techniques to better support the process of annotating reflowable documents by exploiting their properties to flexibly make space to jot down insights and comments close to the target of annotations.

Using ink to transition between interacting with data and externalizing thoughts

The work presented here was published at CHI 2019 [136]. I led the project and I was responsible for the design and implementation of prototypes, designing and conducting the studies. The paper was written collaboratively by all authors.

Pen and touch technology with high-resolution displays, such as the Microsoft Surface Studio or Apple iPad Pro, are becoming mainstream. These devices both detect and differentiate touch from pen, offering much potential for knowledge workers. They afford large workspaces to handle multiple charts and documents, and make users to take advantage from their experience with physical pen and paper. Visualization researchers have started to explore this technology in systems tailored for pen and touch interactions (*e.g.*, SketchInsight [93], VizDom [40], DataInk [190]). But these previous efforts have focused much less on how to enable people to externalize their thoughts as they engage in sensemaking activities.

Previous studies showed that a key affordance of analog pen and paper is to externalize one's thoughts and capture fleeting ideas via annotating content and taking notes [102, 110, 134, 154]. Externalizing thoughts has many cognitive and social benefits such as lowering working memory load, supporting idea reformulation, and providing common ground to share insights with others [91]. These affordances are particularly important for sensemaking and data exploration tasks, as people need to keep track of numerous insights found while browsing through the data, which typically involves many visualizations and documents [127].

We contribute ActiveInk, a system that enables people to actively engage and think with data. It allows for a seamless transition between interacting with data and documents, and externalizing thoughts with pen (Figure 4.1). We designed and evaluated two alternative methods of working with ActiveInk. First, *prefix* provides pen modes and bimanual interaction to allow for switching between *ink pens* and *action pens*. Second,

1. USING INK TO TRANSITION BETWEEN INTERACTING WITH DATA AND EXTERNALIZING THOUGHTS

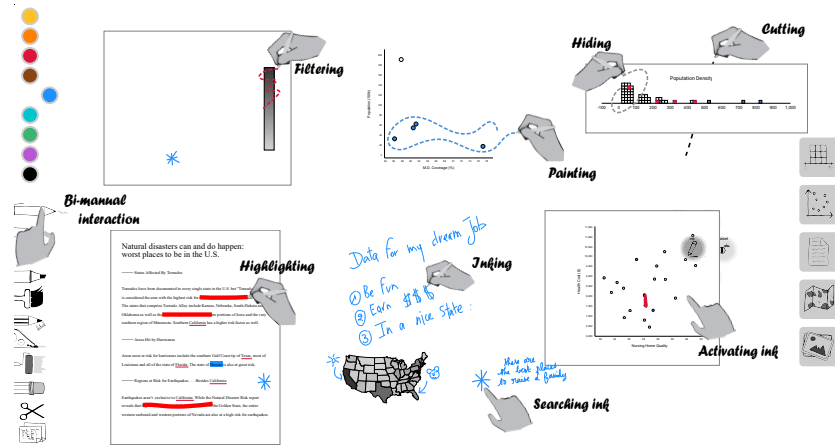


Figure 1.1: ActiveInk affords smooth transition between using a pen for high-precision selections of data and for externalizing thinking via notes and annotations. Ink strokes can be leveraged to perform operations on underlying data.

postfix provides ink for all pen strokes, and enables people to *activate* any ink stroke to perform analytic actions. For example, an annotation made at one point during the exploration process, such as underlining sentences in a document or circling a region in a map, may be activated at any time to filter, highlight, or color the underlying data. All actions propagate to all visualizations and documents in the workspace.

A study comparing these two approaches to a baseline of ink + touch (no actions) revealed benefits and drawbacks to each approach. We also observed that annotating data and taking ink notes were a common part of the analysis process across all conditions, but that ink actions, when available, supplanted some forms of annotation in active reading.

Informed by the study results, we expanded ActiveInk with a hybrid interface, combining both *prefix* and *postfix* approaches. This interface provides both *specific pens* and a *magic ink* pen, whose resulting ink strokes can be activated later on. We enhanced the analytic capabilities of ActiveInk to provide new functions such as activating a sketched correlation line to trigger a regression or hatching an interval on an axis or legend to filter corresponding values. The resulting version of ActiveInk recognizes handwritten content as people take notes, allowing for words and shapes to be searched, filtered, and retrieved to facilitate revisitation.

In summary, our work contributes the following:

1. A novel set of pen-enabled actions for interacting with visualizations, documents, and images in a coordinated workspace;
2. A study on a task showing that users rely on these actions when available for exploratory data analysis, and that both *prefix* and *postfix* have benefits;
3. A hybrid prototype demonstrating how both approaches can be combined in the same environment.

These contributions bring people closer to their data, enabling them to interweave reading and acting on data. ActiveInk provides for seamless annotation, note-taking, and invocation of analytical operations, without even putting the pen down.

1.1 Related Work

This research sits at the intersection of information visualization and human-computer interaction, building on previous designs of pen and touch interfaces for visualization, general digital pen approaches for annotation and note-taking, and findings about the role of externalizing knowledge in solving data analysis tasks.

1.1.1 Pen and Touch Visualization Systems

Visualization researchers have recognized the potential of leveraging input modalities beyond mouse and keyboard in the past decade [92]. Advances in display technology and, in particular, the ability to offer simultaneous pen and touch on high-resolution displays have made possible new approaches to interact with visualizations.

There have been several approaches to design a coherent set of touch gestures for interacting with visualizations on tablets [45, 85, 142, 143, 144], but these do not fully take advantage of the different inking capabilities. Visualization systems such as SketchInsight [93], PanoramicData [196], and VizDom [40] offer novel experiences leveraging pen and touch interactions for data analysis tasks. The research focus of these systems is on the design of interactions to create and interact with visualizations. Pen interactions often support two modes: free-form note-taking and predefined gestures to invoke commands. For example, sketching a particular shape (*e.g.*, a circle) creates a specific type of visualization (*e.g.*, a pie chart) in SketchInsight; drawing a line between two visualizations activates coordination through brushing and linking in PanoramicData. These interactions require learning and remembering a potentially large set of gestures. Although free-form inking supported by these systems enables analysts to externalize their thoughts as they interact with the data, none of these ink marks are leveraged for further interaction.

A recent system, DataInk [190], enables people to sketch arbitrary shapes on a canvas and bind their visual properties (color, size, thickness) to data. While this research focused on creating expressive visualizations for storytelling and art purposes, it demonstrates one way to leverage the unique strengths of using a digital pen for visualizing data. In this work, we do not address data visualization authoring, but rather focus on interaction with existing content.

1.1.2 Unique Affordances of Digital Pens

A recent study by Riche *et al.* [134] compiled and compared activities of people with an analog pen and using mainstream digital pen devices. They extracted a set of pen use cases, several of which are unique to digital pens. A key property of digital pens for working with visualizations is the accuracy for direct pointing and dragging, providing the ability to compose precise selections by sketching complex shapes on top of visualizations, while reducing the occlusion caused by touch. Digital pen systems have other capabilities beyond analog pens, most notably the ability for the same pen to perform different functions in quick succession[77].

Multiple use cases of an analog pen also appear to transfer to a digital pen [134] such as externalizing thoughts and capturing fleeting ideas. Research in human-computer interaction studied and leveraged these phenomena, in particular, for making annotations during active reading of documents [106, 110, 151], drawing,[188, 189] taking notes [76], gathering information [74, 169], generating ideas [95], and sharing ideas with others [198]. However, we are not aware of works which specifically focus on studying and supporting externalization using a pen for data analysis and sensemaking scenarios.

1.1.3 Externalization in Visualization

Externalizing thoughts through annotations on content, taking notes, and drawing diagrams has multiple benefits when engaging in long and complex tasks [91]. Externalizing thoughts enables people to limit their working memory load [148], articulate and reformulate thoughts which can lead to substantial improvements in understanding and retention [116], or share their thinking with peers to generate a shared object of thoughts to support communication and decision-making [34]. Indeed, Kidd hints that the act of annotation itself creates knowledge in the knowledge worker, which may be more important to the process than the marks that remain on the page [88].

These benefits are particularly relevant for visual analytics and sensemaking activities. The sensemaking process is a loop [127] involving phases of analysis and interpretation of visualizations and documents to gain insights, and phases of revisitation to compare and contrast these insights to generate and investigate hypotheses. Externalizing thoughts is critical to facilitate revisitation and keep track of insights and hypotheses, especially over long periods of time.

Recent research started to study active reading of visualizations. Walny *et al.* [180] conducted a series of studies on active reading with both printed and digital visualizations. They demonstrated that active reading behaviors previously observed on textual documents (*e.g.*, underlining, writing notes in margin) occurred when people interpreted visualizations, and that such behaviors transferred in the digital world. While they did not consider complex sensemaking activities involving multiple visualizations and documents, their results suggest that inking with a pen on or around visualizations is an activity that people naturally do when working with visualizations (and given the ability to do so). This work builds upon their findings and reveals externalization strategies people employ during sensemaking activities using a digital pen on interactive data visualizations.

1.2 ActiveInk

Our goal with ActiveInk is to empower people to think with data. We aim at leveraging the digital pen to enable people to interweave *high-precision interactions* performed on visualizations and documents with *externalizing their thoughts* during the exploration process, as they would do with a regular pen on documents and blank paper. In the following, we define *note* as ink separated from data views, *annotations* as ink on data views and *actions* as ink that changes the visual appearance of data.

ActiveInk does not target expert data analysts as they would likely require extensive computations, analytic functions, and the ability to handle large amounts of data (better

supported by systems such as Vizdom [40]). Rather, we seek to exemplify a more fluid sensemaking experience than what exists today for knowledge workers and data enthusiasts. We designed ActiveInk to primarily support thinking as people make sense of their data. We describe below the general principles that guided our design, and ActiveInk's key interface components.

1.2.1 Design Principles

We followed three driving principles to maximize opportunities for people to externalize thoughts while retaining the ability to interact with visualizations, minimizing the physical and cognitive costs of switching between the two.

D1. Support sensemaking through interaction with a set of heterogeneous information.

While many visualization systems focus on interacting with data visualizations, our goal is to support the broader activity of sensemaking, integrating documents and images as well. This activity involves extracting and cross-referencing information from multiple sources. Thus, ActiveInk should provide people with space to think [4], *i.e.*, support viewing multiple of these assets at once and enable individual spatial organizations, where space can be given meaning.

D2. Use the digital pen for interacting with data and for externalizing thoughts.

A pen is more precise than a finger for fine positioning and free-form drawing. Composing a selection in a visualization often relates to drawing as it involves the creation of complex arbitrary shape enclosing a set of points while avoiding others. Having the pen already in hand also increases the opportunities for externalizing thoughts (annotation, note-taking). Thus, we designed a set of pen actions to enable analysis. To reduce the cost of switching between analysis and externalization, we propose two interface strategies (Figure 1.2), which operate on an infinite canvas that we detailed below:

1. *prefix*: select an action to associate with the pen through touch first, and then ink the scope for this action with the pen. Contrary to [191, 27, 124] that consider touch and pen actions performed in parallel with two hands, *prefix* consists of sequential touch and pen actions (potentially but not necessarily performed using two hands).
2. *postfix*: the pen always lays ink on the screen. Tapping an ink stroke with a finger reveals a marking menu of actions to perform on the data associated with that stroke. This *a posteriori* activation of ink strokes takes inspiration from the Scriboli system [73].

D3. Avoid requiring memorization of gestures.

Previous research on active reading and annotation has shown that the form of annotations varies between individuals, and even across tasks within individuals. A circle in one place may not serve the same annotative function as a circle in another, though similar annotations in spatio-temporal proximity often have the same intent [110]. Learning a collection of gestures may be difficult and recalling them may disrupt the flow of annotation. Instead, we aim to provide complete freedom of pen inking, and to support actions that users can *a posteriori* associate with any ink stroke.

1. USING INK TO TRANSITION BETWEEN INTERACTING WITH DATA AND EXTERNALIZING THOUGHTS

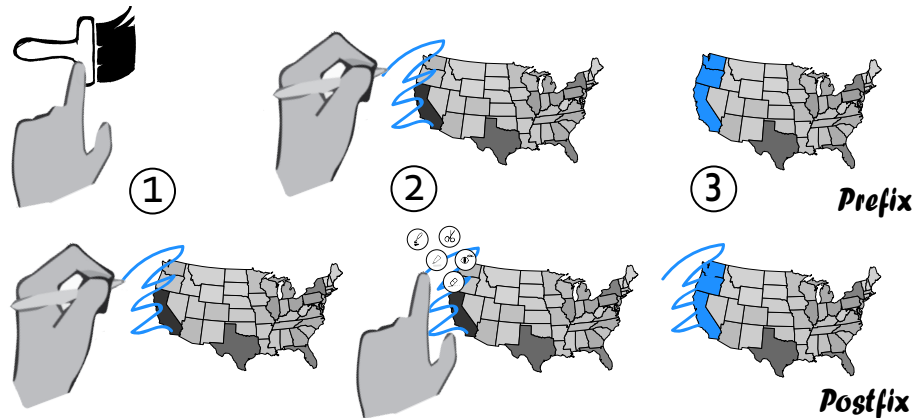


Figure 1.2: Different sequences of interactions to perform an action such as paint on the data underlying the ink stroke.

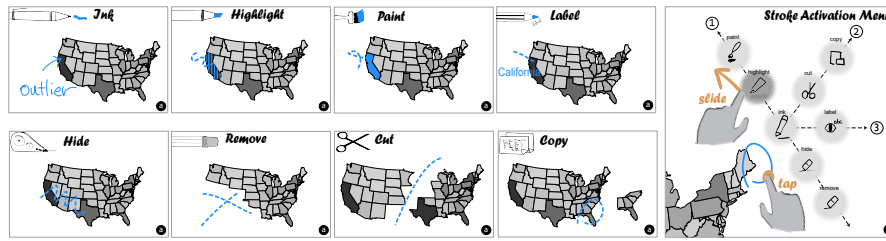


Figure 1.3: Visual effects produced by our 8 different types of pens in *prefix* (a). In *postfix* (b), tapping an ink stroke reveals the activation menu. Sliding a finger from the center to an action previews its effect, while lifting the finger triggers it.

1.2.2 Unified Infinite Canvas

ActiveInk is built around a zoomable infinite *canvas*, on which people can drop visualizations, documents, and images, which we collectively refer to as *views*, and rearrange them. The ability to lay out the workspace, reorganize and manipulate views was previously shown to be an important component of active reading of visualizations [180].

Freeform inking is provided anywhere on the canvas and on views alike. Ink on a view stays with the view when moved. Interaction with the canvas and views uses two-finger pinch to zoom. Single finger slide moves views or pans the canvas.

To provide powerful analysis across visualizations, the views imported into the ActiveInk canvas are always tightly coupled. Each view responds to operations such as filter or highlight performed on the others. To maximize usability and minimize the cognitive effort of tracking coordination, we chose to coordinate all views of the data on the canvas. The data is loaded from tabular data file formats such as CSV, and is stored in-memory as a JSON data structure. In this first prototype we do not use a visualization toolkit, but rather rely on a set of basic, common chart types implemented from scratch in our environment for the sake of easy customization. ActiveInk currently supports binned histograms, scatterplots, maps, and text documents. We selected this subset

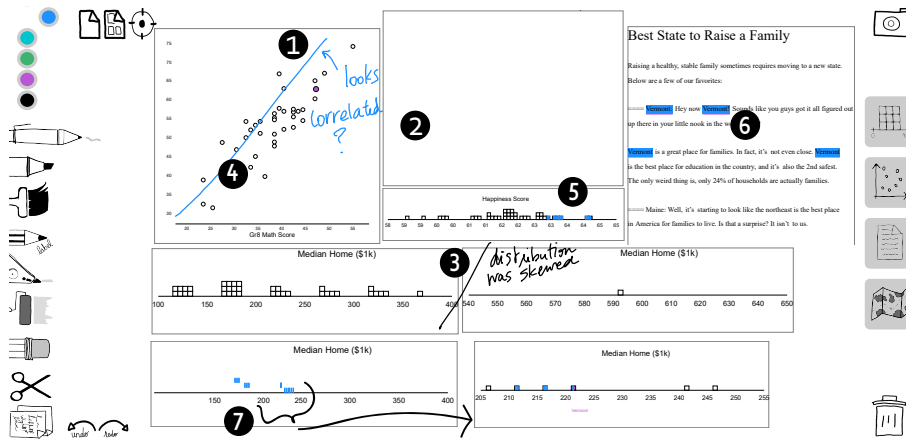


















Figure 1.4: Depiction of the canvas of a data journalist organizing their workspace and annotating findings (1-ink , 2-cut , 3-remove ) and interacting on the data to identify salient subsets (4-hide , 5-highlight , 6-paint  and label , and 7-copy  a subset of a histogram).

for the diversity of data, interactions, and insights they can offer. However, adding additional types of visualizations such as line charts or density plots is straightforward.

1.2.3 Pen Actions

ActiveInk supports the set of operations on views illustrated in Figure 1.3. The following scenario, illustrated in Figure 1.4, describes the pen actions of ActiveInk and how one can use these operations for sensemaking using the *prefix* strategy.

Emma is a data journalist writing a piece about the best states in the U.S. to start a family. She has collected data about each state, such as school quality, air pollution, and median home price, as well as documents about this topic. She organizes the most interesting views in her canvas, moving them with a finger and pinching them to resize. She annotates salient insights with *ink*  as she encounters them. For example, she notes that the school quality index mostly correlates with math scores in Grade 8. She uses *cut*  to slice the binned histogram of house prices, as the distribution is skewed. Providing a healthy environment being critical, she decides to remove all states that have higher air pollution by switching to the *remove*  pen and using it on dark regions of the map. Data is removed from the entire canvas, and the scales in scatterplots and histograms are updated in response.

Since school quality is an important factor to raise a child, Emma opts to *hide*  the lower quadrant of the scatterplot, deemphasizing those states in all views. She then chooses a blue color to *highlight*  the higher portion of happiness score. She notices that one of them, Vermont, pops up in the document and thus reads the corresponding paragraph. To identify all data about Vermont, she switches to a purple color to *paint*  and *label*  it from the text. Since Emma wants to have more details on house prices in Vermont, she uses the *copy*  instrument and lassos the interval containing Vermont. She sees Vermont is in the lower range, concluding that it is the place to be!

1.2.4 Prefix Interface: Switching between Types of Pen

The *prefix* interface offers the pen actions described above as different types of instruments people can acquire with the pen (Figure 1.3). This paradigm of different *modes of interaction* is usually found in mainstream interfaces today, although modes generally appear in a menu or ribbon positioned at the top. In contrast, ActiveInk presents it on the non-dominant hand side of the workspace to make users able to select (tap) modes with the thumb of their non-dominant hand, and then use the pen with their other hand. The selected mode remains active until users select another one. This design may lower the cost of interaction for mode switching [6].








The ActiveInk *prefix* interface uses a paradigm familiar to most people and provides a comprehensive view of all actions at all times, which is important for non-experts [149]. This paradigm is also efficient for accomplishing actions in batches. However, people need to decide on mode before performing actions, and interweaving different types of activities may require numerous mode switches. Also, contrary to quasimodes [130], it avoids maintaining tense postures but with the drawback of potentially forgetting what mode is currently activated.

Forgetting to switch modes may result in mistakes to undo later (*e.g.*, people want to rapidly jot down thoughts next to highlighted points in a scatterplot but the pen highlights more points instead). Action pens in *prefix* lay down dashed ink to mitigate mode errors, and the interface includes buttons for global undo and redo to correct mistakes. The *clear* pen ✕ is used to clear all visual properties of data elements resulting from prior interactions (highlight, paint, *etc.*), affording some level of local undo.

1.2.5 Postfix Interface: Activating Ink Strokes

The *postfix* interface offers the pen actions described above through an in-place menu invoked by tapping an ink stroke (Figure 1.3, right). Activated ink strokes remain on canvas and can be deactivated to undo the action. We hypothesized that this design would stimulate externalizations as the pen always lays down ink first without requiring to switch back to this mode after interacting with the data.

Inspired by the design of the ZoomCatcher menu [191] and recommendation about Multi-Stroke Marking Menus [73, 197], we created a marking menu and organized the actions along three semantic axes:

- 1 **Visual saliency** of data elements that one can increase with *Highlight*  and *Paint* , and decrease with *Hide*  and ultimately *Remove* .
- 2 **Display data** about the underlying elements with *Label*  displaying the name of data entities and their values in histograms.
- 3 **View manipulations** that do neither affect the data nor propagate to other views, including *cut*  and *copy* .

Once the menu is invoked by tapping a stroke, one can preview the effects of any action before executing it by sliding their finger from the central ink action to any other menu item. Lifting the finger triggers the action and fades out the ink stroke to limit visual clutter. Note that actions that are unavailable (*e.g.*, due to no underlying data to highlight) are greyed out. Erasing an activated (faded-out) stroke undoes the action and

reverts the stroke to a regular ink (that can potentially be activated again to perform an action using the menu, or fully erased). This mechanism allows for local undo of actions during the exploration.

The *postfix* interface uses a paradigm that is less familiar to people, and certainly has a steeper learning curve, as all actions are not visible at all times. Given the additional cost of tapping after each ink stroke, this paradigm may slow down actions performed in batches. However, it enables the smooth interweaving of interactions with data and ink externalizations without mode switch (*e.g.*, paint a set of points and hand write the reason for this action next to them). Additionally, actions can be carried out at any time: an underline, arrow, or circle made early during an initial read of the data can later be activated by tapping, thus turning externalizations into potential interaction locations. Since activated strokes remain on the canvas, we also hypothesized that they would provide additional benefits for analytic provenance — assist people in recalling their process to gain insights on the data. Finally, the *postfix* interface has fewer tool icons, and thus is less cluttered.

1.2.6 Implementation

ActiveInk is built as a web-based application in JavaScript and runs on a NodeJS server. The vector graphics canvas is implemented using Paper.js [94] and the visualizations are created with d3.js [25] and BubbleSets.js [37]. To best associate ink strokes with data, ActiveInk tests for data item enclosure or intersection. This allows for marks such as circles and arrows to appropriately associate with the intended data.

1.3 Qualitative Study

1.3.1 Hardware Capabilities

As hardware capabilities might impact the externalization process and users' experience with digital pens, we scope our research questions and results to the hardware device that we considered. For this study, we used a Microsoft Surface Studio [114], with a 28" screen of resolution 4500×3000 pixels, supporting multitouch and pen input.

1.3.2 Research questions

While previous work has investigated active reading of simple graph visualizations [180], it is not yet clear how to support externalization for data analysis, and complex sensemaking scenarios that involve multiple visualizations and documents.

To gain insights on the externalization process during sensemaking using a digital pen for data analysis, we conducted a qualitative observation study. We designed the procedure to investigate the following research questions:

Q1. How do externalizations manifest during sensemaking with digital inking when working with multiple visualizations?

Our first question is whether results from Walny *et al.* [180] will transfer to more complex tasks involving extracting and comparing insights from multiple visualizations and documents to make sense of the data. We seek to observe how people organize their findings and extract information from multiple sources using an infinite canvas.

1. USING INK TO TRANSITION BETWEEN INTERACTING WITH DATA AND EXTERNALIZING THOUGHTS

Q2. How do externalizations manifest during sensemaking with ActiveInk?

We seek to observe if and how annotation and note-taking behaviors change when offered the possibility to interact with the data to visually encode it (*e.g.*, highlight). In particular, a key question is whether the cost of switching mode between interaction with data (*e.g.*, highlight, filter) and inking in *prefix* may discourage spontaneous externalization of thinking. In contrast, while *postfix* may encourage such externalization, it uses a less familiar interaction paradigm where actions are not visible at all time, which may prove more difficult to use.

Study material and screenshots of canvases created by participants are available as supplemental material.

1.3.3 Study Design

We used a within-participants design as prior work on active-reading [102, 110, 180] indicated that individual variability would be high given the diverse range of strategies for sensemaking and behaviors for externalization. Thus, participants experienced the following three interfaces in a single session, on a Microsoft Surface Studio using our web-based prototype.

1.3.3.1 Ink

The condition explored sensemaking using only digital ink and touch. Participants could browse through a set of views (visualizations and documents), adding and organizing them in the workspace with touch. To inspect underlying data, participants could tap and hold with a finger on a visual element to display its label and data values in all views. Inking was available anywhere on the canvas, including on views. This condition served as our baseline, enabling us to build knowledge on active reading and externalization behaviors without any analytic functions linked to pen interactions.

1.3.3.2 ActiveInk Prefix and Postfix

In the second and third conditions, participants used ActiveInk with *prefix* and *postfix* interfaces, as described above. Inking and touch capabilities were also provided as in the ink condition. As we hypothesized an order effect (due to fatigue or learning), we counterbalanced the presentation order of these two conditions. Participants also had to complete different scenarios in each of these two conditions. Assignment between a scenario and a condition was counterbalanced across participants in order to avoid observing an effect of the dataset instead of observing an effect of the ink condition.

1.3.4 Participants and Method

We recruited 8 participants (2 females, 6 males; 7 in 30s and 1 in 20s) from a large software company (over 60000 employees), screening them in to cover different age ranges, genders, and roles (team managers, administrative assistants, developers, data scientists, researchers). We screened participants for at least one year of experience with a pen-enabled device and recent experience reading and creating simple charts from data. Only P5 reported data analysis is not part of his/her job. We piloted the study with 2 people to streamline the training and fix minor usability issues.

Participants were encouraged to adjust the orientation and incline of the device for comfortable pen and touch interactions. The study was held in a quiet room; experimenters observed live, from behind a space divider, using a camera.

1.3.4.1 Data and Task

As our goal was to observe externalization during sensemaking, we designed high-level tasks involving browsing through a set of views, making hypotheses, and investigating queries to understand the data. We prepared 3 similar analytic scenarios based on identifying a subset of states in the U.S. In the ink condition, participants searched for states where tobacco and alcohol use was correlated to accidents and overdoses. In the *prefix* and *postfix* conditions we counterbalanced across two scenarios: identifying the best and worst states to retire, or start a new job and raise a family. A different multivariate dataset was provided for each scenario, containing factors such as nursing home quality (retirement) and home prices (new job). Examples appear in Figures 4.1 and 1.4. Participants were instructed to imagine they were conducting research to write an article about their findings, and should create a canvas that would enable them to recall these findings in several weeks. To limit the duration of the session, we selected a subset of visualizations and documents (9 histograms, 9 scatterplots, 3 documents, and 3 maps) for each exploration task.

1.3.4.2 Procedure

An experimenter first instructed participants to sign a consent form and fill out a demographics questionnaire. The study was structured in three phases, one per condition. For each condition, the experimenter briefly described the interface and dataset at the start of the training. To cover the interaction mechanics, the experimenter instructed participants to perform each action available in the interface (pen and touch), explaining the interaction and resulting effects, and informing participants of efficient interaction techniques if needed (*e.g.*, bi-manual interactions). This training lasted about 10 minutes. Since ActiveInk features a large number of actions, these conditions included an additional 5 minute practice task, in which participants completed a brief analysis task while asking questions and receiving experimenter guidance as needed.

After training, participants completed the main data exploration task for a maximum of 15 minutes. After each task, participants completed a questionnaire and had a brief interview with the experimenter about their likes, dislikes, and wishes. Participants were instructed to take a 5 minute break after the second condition. After experiencing all conditions, participants answered a preference questionnaire and verbally explained the advantages and drawbacks of each technique. The entire experiment lasted two hours. Participants received a \$150 gift card as compensation.

1.3.4.3 Data Collection & Data Analysis

We logged pen interactions with ActiveInk, and used a screen video capture and video-audio recordings. At least two experimenters were present for each session: one person interacted with the participant while the other took notes.

We then gathered pen interactions, analyzed and plotted them using the PowerBI tool into a color coded diagram (Figure 1.5). We also gathered and categorized feedback we got from participants during the study into the following categories: positive feedback,

1. USING INK TO TRANSITION BETWEEN INTERACTING WITH DATA AND EXTERNALIZING THOUGHTS

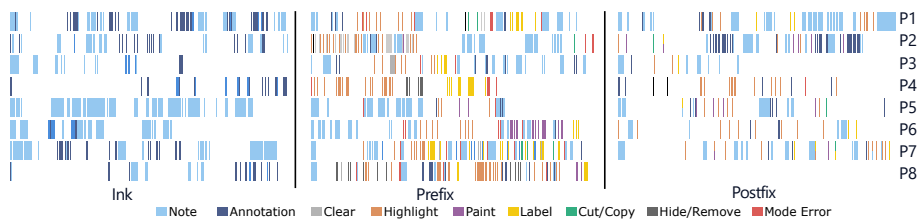


Figure 1.5: Inking and action sequences per participant across conditions. Each block reflects a 15 minute analysis task. Participants carried out the most annotations in the ink condition, followed by *postfix*. The most common operations were highlight, paint, and label, which provide details on the data. Split bars in *postfix* indicate ink strokes used for multiple actions. Note that P8-*postfix* logs were not saved due to a technical error.

negative feedback, problems they encountered with the interface and wishes about additional functionality. Finally, we cross-referenced users' observations, feedback from the study and answers from the reference questionnaires to better understand their sense-making strategies and preferences using either postfix or prefix.

1.3.5 Results

Figure 1.5 depicts the interactions of each participant over the course of each exploration task. All canvases created in the study are available at: <https://hugoromat.github.io/ActiveInk/> and in Appendix A.2.

1.3.5.1 Ink

In the first condition, six participants used inking extensively for making sense of data (Figure 1.5). The largest portion of ink across participants are *notes* separated from data (78.0% of strokes), as opposed to direct *annotations* on views (22.0%). Participants exhibited different behaviors: P1 and P5 laid down most ink, however, P1 made both notes and annotations while P5 only took notes. While P4 and P8 did not ink as much as others, their ink mostly consisted of annotations.

Figure 1.6 depicts externalizations in the ink-only condition. It is interesting to note that two participants (P1,P4 in Figure 1.6) used inking to coordinate multiple subsets of data permanently on the canvas. These behaviors correspond to the paint, highlight, and label actions of ActiveInk. We also observed the depiction of hypotheses about the data (*e.g.*, drawing a correlation line) and externalized queries (*e.g.*, drawing a threshold). Both of these annotations are illustrated in Figure 1.6-P1.

Most of the notes on the canvas were phrases summarizing findings, hypotheses, or describing the sensemaking process itself. However, we also saw instances of pictorial representations such as arrows (6 participants), or lines on the canvas to divide the workspace into regions (P3, P5). We also observed participants visually link a pictorial annotation in a visualization to a separate handwritten note. For example, P1 drew a box in red around the word “threshold” referencing the line sketched with the same pen in both scatterplots (Figure 1.6-P1). Other participants (P1, P2, P6, P7) used different colors, visually connecting notes to relevant annotations.

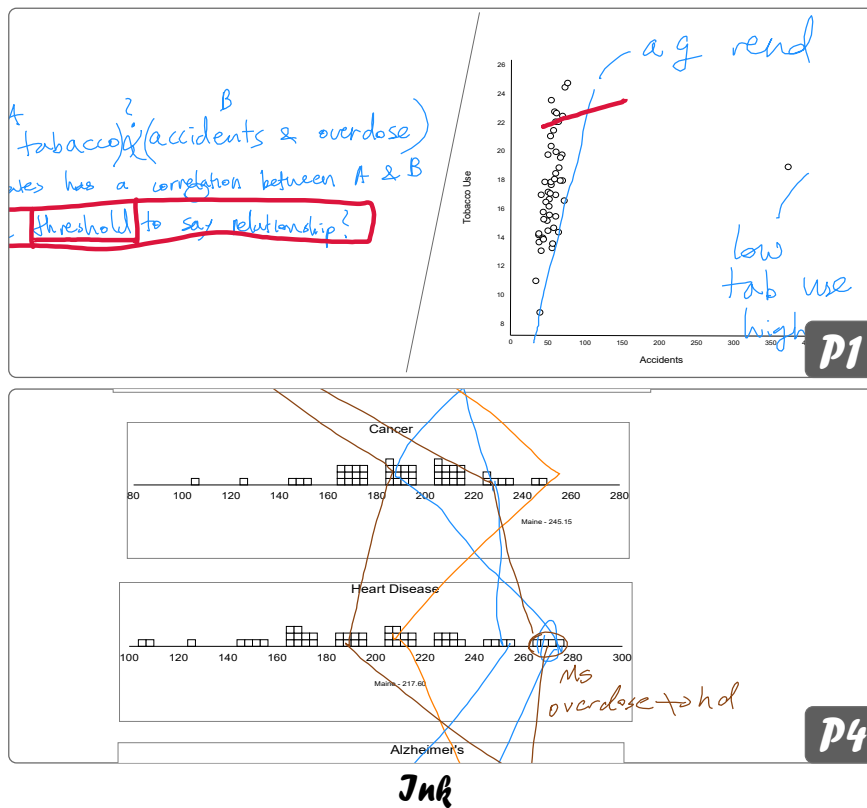


Figure 1.6: Examples of participants' externalizations during sensemaking with ink. P1 used color to connect notes and annotations, while P4 drew lines to connect data items across views.

1.3.5.2 Externalizations with ActiveInk

In contrast to the ink-only condition, the amount of regular ink in both ActiveInk conditions decreased, with fewer annotations. The prevalence of notes did not decrease as much: 8% lower in *prefix* and slightly higher in *postfix*. The total number of annotation + action strokes was consistently between 19–26% across all conditions. This may indicate that actions serve the role of some annotations in ActiveInk.

While there were fewer annotations than in the ink condition, we noted several interesting types of annotations created after operating on the data: (1) depicting how views relate after applying cut or copy (*e.g.*, copying a subset of a view), (2) depicting insights (*e.g.*, correlation between two data dimensions), (3) identifying data of interest (*e.g.*, circling), and (4) characterizing data of interest (*e.g.*, good vs. bad candidates). In the ink condition, we saw annotations to connect specific data points across multiple views. We did not observe this type of annotation with ActiveInk since actions such as highlight and paint fulfilled this role.

The ink stroke log analysis (Table 1.1) indicates that *postfix* encourages more annotation than *prefix*. In observing the use of *prefix*, we saw that participants chose a pen and

1. USING INK TO TRANSITION BETWEEN INTERACTING WITH DATA AND EXTERNALIZING THOUGHTS

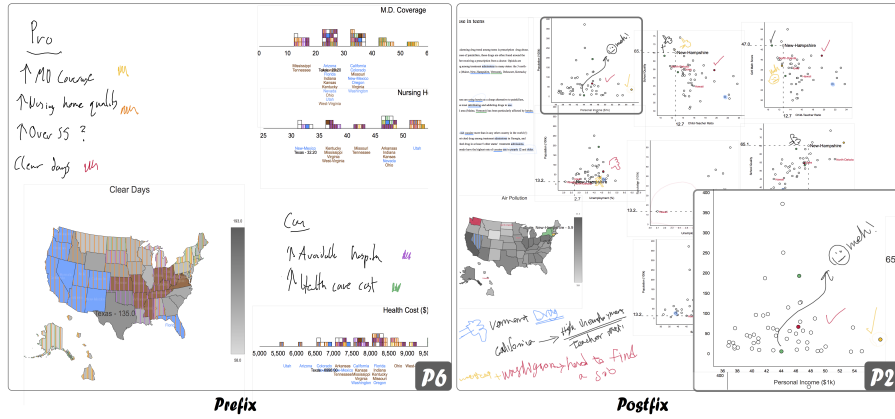


Figure 1.7: Examples of participants' externalizations during sensemaking with ActiveInk, including batch highlighting with *prefix* on the left, and interwoven actions and note-taking with colors using *postfix* on the right.

Stroke Type	Ink	<i>prefix</i>	<i>postfix</i>
Notes	1082	78.0%	783 80.2%
Annotations	305	22.0%	59 4.2% 98 10.1%
Actions	0	314 22.4%	89 9.1%
Errors	0	46 3.3%	6 0.6%
Total	1387	100%	1402 100% 976 100%

Table 1.1: Ink stroke analysis. Errors are data action on empty canvas.

repeatedly made short strokes on items one by one, whereas with *postfix* they more commonly used enclosures to select groups of items, then activated the ink.

The most frequent action performed across both ActiveInk conditions was highlight (187), followed by label (55), paint (50), filter (30), cut (6), copy (3), hide (3), and remove (2). Video analysis shows that cut and remove were primarily used on skewed distributions and outliers in scatterplots and histograms. In the *prefix* condition, the clear function was used to remove actions 47 times, and undo was used 39 times. Undo and clear were not provided in *postfix* as ink actions were locally reversible at all times. All participants commented that ActiveInk analytic functions on data were useful in contrast to the ink condition.

While a higher fraction of strokes were used for note-taking in *postfix*, the note-taking behavior was consistent in both ActiveInk conditions. For example, P7 wrote hypotheses on top of the canvas, interacted with data visualizations to investigate them, and wrote answers next to original notes in both conditions. Similar to the ink condition, participants created colored legends (e.g., Figure 1.7-P2) in both ActiveInk conditions. Overall, we observed meaning assigned to color in 75% of all exploration tasks across participants and conditions. We observed that participants generally followed one or more of these strategies: 1) write the data dimension name or value, 2) use the same color to mark the data subset, and 3) use a pictorial representation (e.g., underline, check mark) to connect notes in one place to data in different visualizations.

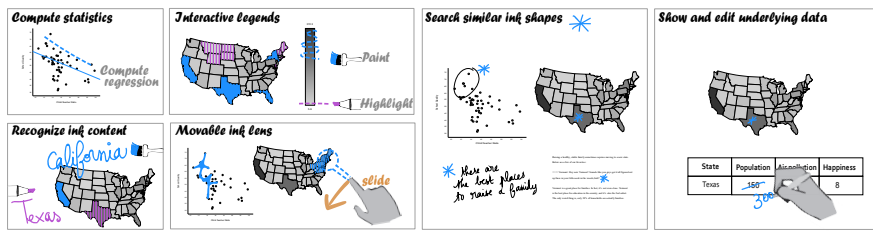


Figure 1.8: Advanced actions added to ActiveInk, informed by results of our qualitative study.

1.3.5.3 Sensemaking strategies with ActiveInk

We observed two different strategies that reflect advantages of each interface (Figure 1.7). P2 (Figure 1.7-P2) seamlessly interweaved paint action on data item, and in-place annotation with the same color to create a pictorial legend about this data and value. He sketched glyphs such as check marks and thumbs up to indicate positive aspects, and sad faces and thumbs down to indicate negative aspects. This strategy worked well with *postfix* as the pen “automatically” switched to ink after performing an action on the data. In contrast, P6’s strategy was to do actions in batches (Figure 1.7-P6). Using the ink pen, P6 first handwrote a legend to associate certain data values to positives and negatives, then switched to the highlight pen to perform highlights in batches, only switching colors. This strategy worked well with *prefix*, as it did not require tapping strokes, but rather pick a single type of pen to apply successive strokes on the views.

We observed differences between participants but each followed a personal sensemaking strategy, generally remaining consistent across conditions (Figure 1.5). For example, P1 used inking extensively in the first condition and continued to take more notes than others in *prefix* and *postfix* while most of the annotations were replaced by pen actions. In contrast, P4 mostly relied on annotations in ink condition, which were replaced by pen actions (mostly highlight) in ActiveInk.

Most participants (6/8) attempted to lay out visualizations and documents on the canvas so they were all in view without requiring zooming or panning. P7 made use of the larger surface, storing handwritten notes on hypotheses and results out of screen. Two participants used a lower corner to place temporary visualizations. P8 used the histograms as one would use dynamic queries in conventional coordinated view interfaces, as a way to filter the data according to different attributes. P8 is also the only one who leveraged multiple tabs to segment the workspace by different questions investigated (e.g., one canvas for identifying the best states where to retire, another for the worst ones).

1.3.5.4 Ratings and Preferences

Participants praised the quality of both interfaces. For example, P1 said “*I really like it, it’s the optimal for this type of task really.*” (referring to the general principle of ActiveInk, independently of *prefix* or *postfix*). Note that while our questionnaire did not have an option to rate as equal, three participants verbally indicated that they did not have a clear favorite. Overall, six participants preferred *prefix* and two preferred

1. USING INK TO TRANSITION BETWEEN INTERACTING WITH DATA AND EXTERNALIZING THOUGHTS

postfix. In *prefix*, six participants used bimanual interaction, five of whom preferred *prefix* over *postfix* overall.

Results of the subjective experience questionnaire administered after each condition revealed similar ratings for all conditions, although *postfix* rated slightly higher than *prefix* on most measures. Most (6/8 in *prefix* and 8/8 in *postfix*) felt efficient with ActiveInk. Participants made more notes and annotations with *postfix*, while with *prefix*, they carried out more actions. In *prefix*, at least 3.3% were mode errors (actions on the canvas) (Table 1.1). Although, note that we could not detect mode errors resulting from accidental inking. Four participants specifically mentioned mode errors as frustrating, e.g., P3: “I keep just forgetting which one I’m in,” P7: “When I was highlighting and wanted to go back to my notes I forgot to click here.”

In *postfix*, the menu was activated without previewing or applying any action 6 times, which we also counted as errors. We also observed three participants experiencing usability issues with the *postfix* menu. In particular, P1 said: “I would like something faster, like tap and choose rather than tap, wait for animation, and drag to choose.” While dragging on the menu was intended to preview action results before executing them, these previews were not frequently used (only six instances from three participants). Several requested to see all available actions at once. This may be especially important during the learning period as participants are not familiar with the semantics of the actions available and thus do not fully comprehend their organization in the menu.

1.3.5.5 Wishes

Three participants mentioned the potential of a hybrid interface, integrating the best of both approaches. P3 said, “Can you make something where some people can choose the tool first and other people can select things and change them?” while P9 suggested using physical buttons on the pen to activate the most common actions. P5 suggested the ability to see the underlying data in a table, and to be able to change data values directly in the interface. P8 commented on the need for transient pen highlights “I would like to drag my ink across the thing and ‘boom’ things light up.” In addition to local undo provided by erasing an activated ink in *postfix*, P2 and P6 suggested a global undo button would also be helpful. P2 and P6 also suggested functions to align or snap views together for neater canvases. Finally, both P1 and P5 suggested sketching dividing lines on the canvas to create different sets of coordinated views (in addition to different tabs).

1.3.6 Summary and Discussion

Our results suggest that the results of Walny et al. on active reading of one visualization [180] transfer to sensemaking tasks involving multiple visualizations and documents. When ink actions are available, people favor visually marking the data using actions rather than annotating with regular ink, though note-taking was present in all conditions. We observed that several annotations were used to add information not built into the action tools (e.g., sketched glyphs, semantic use of color). Annotations were reused by all participants in describing their findings and strategies to the experimenter, though we did not specifically test for the benefit of inking for recall. While we believe there are sensemaking benefits resulting from the low cost of externalizing thoughts with digital inking, more investigation is needed to

check whether this behavior also happens in conventional systems using mouse and keyboards.

Most participants used notes, physically separated from the data, to record their questions and hypotheses, and evidence to confirm or reject them. Participants linked their handwritten notes to data by either rewriting the data dimension name or value, using the same ink color as used for data highlights, or using a pictorial representation (*e.g.*, a check mark, an arrow). These techniques reflect the creation of ad-hoc visual indices to connect and cross-reference related information from different sources. This manual and iterative process may explain why most participants attempted to keep everything in one view, and suggests the need to better support management of large canvases or multiple pages of information.

1.3.7 Implications for Design

As evidenced by the ink stroke analysis, the *prefix* and *postfix* interfaces lead to different behaviors: *prefix* supported rapid sequential actions, while *postfix* better integrated with annotation and note-taking. This implies a better solution would be to support both paradigms. Thus, we created a hybrid version which offers the same set of pens available in the *prefix* interface, as well as the ability to activate strokes laid down with the ink pen and marker using the *postfix* activation menu. As before, activated ink strokes remain on the canvas, constituting a visual trace of the action performed, and a means to undo the action later on.

The observations we made about note-taking during sensemaking with ActiveInk prompted us to add a 4th design principle:

D4. Make externalizations useful.

We observed that, as people explore data, they took notes separated from the data visualizations or documents to capture their hypotheses or summarize their findings. These ink strokes contained explicit references to data dimensions or values through textual content, color, or associated shape to enable cross-referencing insights in multiple sources (notes, data visualizations, and documents). ActiveInk should leverage these externalizations for further analysis.

1.3.8 Advanced Actions to Support Sensemaking

The result of our qualitative study informed a set of 6 new functions depicted in Figure 1.8, inspired by our observations and comments from our participants.

Compute statistics: Sketching hypotheses about statistical properties of the data directly on the visualization is common (*e.g.*, sketching a correlation line in a scatterplot). While laying ink is a mean to quickly mark potentially important insights discovered serendipitously while exploring the data, getting back to these and assessing their correctness is important when drawing conclusions about the data. For this reason, we added the ability to compute such statistics by tapping on a stroke and selecting the statistics action (*i.e.*, adding a new options in the *postfix* menu). Note that ActiveInk currently supports computing a correlation line in scatterplots and averaging the values of a set of data items in histograms as we mostly observed these behaviors in the study.

Interactive legends: We also observed participants attempting to draw on the map legend (and sometimes tap it with a finger). We incorporated the ability to sketch on

1. USING INK TO TRANSITION BETWEEN INTERACTING WITH DATA AND EXTERNALIZING THOUGHTS

the legend to perform actions (highlight, paint, label, hide, *etc.*) on the corresponding data elements.

Ink lenses: Study participants surfaced the need for transient and interactive pen interactions akin to brushing in conventional data visualization systems. We added the ability to create and activate any set of strokes on empty canvas with functions such as highlight or paint. Moving these activated strokes with a finger over views immediately performs the corresponding action on the underlying data, effectively acting as lenses on the data [173] and acting as dynamic queries [160] when activated with filtering.

Recognizing ink content: Since we saw many references to data dimensions and values in the notes participants took, we added handwriting recognition to ActiveInk. This capability turns handwritten notes into content coordinated with the views. In particular, ActiveInk now offers the ability to tap a written word and perform an action on related data. This capability is inspired from the behavior we saw participants exhibit when referring to data dimensions or values in their notes to connect them to the data. For example, one would observe states with low unemployment, identifying Texas and California. Highlighting Texas to further investigate its values along other dimensions, they would make a note to get back to California later. ActiveInk facilitates this process as one would simply tap the written word California, select paint from the *postfix* menu and see the corresponding data items in visualizations and text using the same ink color. Note that handwriting recognition is performed with the MS Ink Analysis API [155].

Search similar ink shapes: Since we observed people draw the same glyph on different views, we added the ability to search for similar sets of strokes making up a glyph. When a search is triggered on a glyph, ActiveInk retrieves similar sets of strokes and brings them, and their underlying content, close to the location of the search action. This facilitates cross-referencing and may prove useful for a large canvas. Strokes are associated with glyphs using a spatiotemporal distance function and threshold, while glyphs are associated with each other using the \$N\$-recognizer [5].

Show and edit underlying data: Participants requested the ability to see the underlying data table for a subset of data, and one participant suggested she would like to edit the data directly. We added the *show data* action to reveal the underlying data table for items associated with a pen stroke. When writing numbers in the table, the written values are recognized and the corresponding data items are updated across all views.

1.4 Conclusion and Future Work

ActiveInk represents a foray into active reading applications for heterogeneous data including visualizations, maps, and documents. It blends the operations of externalizing thoughts with analytic actions in a seamless combination of functions to be carried out with digital pen+touch. The hybrid interface of ActiveInk integrates two ways of working with pen actions, which our study revealed to be complementary for sensemaking activities. *Prefix* inking is useful for batch operations, while *postfix* ink activation is useful for interleaving multiple actions with externalizing thoughts.

This work offers several opportunities for extension. The ink recognition system we used does not distinguish between classes of marks, such as enclosures, underlines, and arrows. Our data selection model associates marks with any data items enclosed or intersected by an ink stroke. Distinguishing types of mark could be interesting to implement a finer ink-data association strategy. ActiveInk was designed as a web

application to work on many platforms and devices. However, in our study we have only used a Microsoft Surface. To better understand the externalization process on digital devices, it would be interesting to investigate how the hardware experience might change the way people interact with application designed to work with digital ink.

ActiveInk was made to demonstrate the possibilities of thinking with ink, but does not scale to a large number of views or with very large volumes of data. Addressing scalability issues requires further development, and implementing the ability to save, recall, and share analysis canvases. It also implies adding support for a wider diversity of charts, which could be achieved by moving to, *e.g.*, Vega-lite [146].

To better understand the impact of ActiveInk on sensemaking and externalization, including any potential effects on the depth and quality of insights, we would like to run a longitudinal study in which participants analyze their own data using the hybrid interface. We suspect performance may improve with long-term use, and sensemaking strategies may change as people feel more engaged with their data. Furthermore, it would be interesting to study the use of ActiveInk in collaborative synchronous and asynchronous sensemaking scenarios.

Making space for externalizing thoughts on reflowable documents that support various content

The work presented here was published at UIST 2019 [138]. During this project I was responsible for the design and implementation of prototypes and techniques, and I designed and conducted the studies. The paper was written collaboratively by all authors.

Annotations are central to active reading. They are a means to emphasize and memorize specific pieces of information, to facilitate re-reading, to identify passages to revise and suggest edits, or to support sense-making [121, 102, 194, 134]. When annotating a document, people make ink marks directly on the content: they underline words, they draw links between related items, they circle elements in figures. These *overlay marks* are often accompanied by *in-context annotations*, that typically take the form of handwritten footnotes, marginalia, or sketches. In-context annotations are not overlaid on top of the document's text and figures, but rather placed close to the content that elicited them, wherever there is free space to accommodate them.

On paper, of course, such space is extremely limited and constrained by physical pages. But with digital ink, there is a whole zoo of design choices and trade-offs that are possible. In this work, we seek to articulate some of these options, explore their implications, and thereby sketch out this new design space of digital annotations that are not necessarily constrained by the physical space.

Starting with the first active reading machine introduced in 1998 [151], many research projects have sought to improve active reading on digital devices (*e.g.*, [128, 169, 106]). Together with technological advances in hardware, including multi-touch input and high-resolution pens, digital devices enable people to mark up documents in ways that may go beyond what is possible with physical pen and paper [115]. Digital ink does

2. MAKING SPACE FOR EXTERNALIZING THOUGHTS ON REFLOWABLE DOCUMENTS THAT SUPPORT VARIOUS CONTENT

not dry. Digital annotations can easily be removed, repositioned, or revisited. They can be archived, shared with others, turned into navigation bookmarks [169], and even made active, triggering search and queries on the underlying content [58, 76].

But while annotating a printed document is just about jotting brief comments or ideas and anchoring them to a particular passage (e.g. by call-out lines), annotating an electronic document is often much more cumbersome. People have to create post-it like notes and fill them in, or create comments that will appear in a dedicated sidebar. These interactions with the interface impact active reading as they interfere with the capture of fleeting thoughts, driving many people to instead rely on pen and paper [134].

The goal of our research is to reduce such interface friction and offer fluid interactions for interleaving overlay marks and in-context annotations. We also consider digital documents more broadly than previous research: digital ink may not dry, but the content of electronic documents does not dry either. Document content evolves, most online web pages are non-paginated and render content differently depending on screen sizes. Anchoring in-context annotations at the right location in the document to facilitate re-reading becomes critical. To articulate design choices and trade-offs for in-context annotation, we introduce SpaceInk, a design space of pen+touch input techniques that make room for in-context annotations by dynamically reflowing a document's content. The design space organizes techniques according to *when* additional space is created: before, while or after the user handwrites annotations; and *where* additional space is created: on the paragraph's side, wrapped inside the paragraph, between lines, or between words.

SpaceInk encompasses existing techniques such as TextTearing [193] – enabling the creation of white space between paragraphs in paginated documents to write annotations – and uncovers variations – e.g. space is made *a posteriori* avoiding user's premature commitment. It also suggests several new techniques taking into account **annotations' scope** – whether annotations are anchored to a word, a sentence or a paragraph; and multiple levels of **user agency** – whether users control the location and size of the space needed, the system automatically adjust it as they write or a combination thereof. We gather feedback about these techniques in a user study, the results of which inform the design of a prototype system that lets users concentrate on capturing fleeting thoughts, streamlining the overall annotation input process by enabling the fluid interleaving of space-making and inking actions, not only making space to ink, but also making space to think.

To summarize, our contributions are the following:

- SpaceInk: a design space of pen+touch input techniques for in-context annotations
- Insights from a user study on 5 techniques in this design space
- Design rationale and implementation of a prototype system for fluidly interleaving space-making and inking actions

2.1 Related Work

2.1.1 Digital Active Reading

Researchers from the XLibris project [151, 58, 57] were the first to propose an “active reading machine”, with which users could annotate text using freeform marks on a pen+tablet. The prototype made it possible for handwritten notes to coexist with regular text, offering an experience close to taking notes with an analog pen on a printed document. As opposed to text input with a keyboard, handwriting with a stylus allows readers to write on the material while keeping their marks clearly distinguishable from the original content, a feature that is especially important to readers [121, 102].

Active reading with stylus-equipped devices has been further investigated since then. Matulic and Norrie [106] describe an application for active reading that uses pen+touch input on a tablet. Pen input is dedicated to annotations, while multi-touch gestures support navigation: flipping pages, jumping to a specific page using space-filling thumbnails [36]. PapierCraft [96] tries to bring the best of both worlds into one application, by letting people interact with Paper Augmented Digital Documents allowing an extendable command set. Another example is LiquidText [169, 168], an application that builds upon the XLibris notebook’s design, where users could paste annotated text segments as clippings [57]. LiquidText splits the viewport into two areas: one shows the main document while the other shows a workspace that stores excerpts from it, dropped there by users. This workspace features advanced interactions for grouping annotations, linking them to several parts of the document, and even using them as navigational cues into the paginated document. LiquidText also enables users to collapse portions of a document in the spirit of the Mélange technique [49] to better support the sort of side-by-side comparisons that active readers often make [121, 115].

2.1.2 Active Annotations

As mentioned earlier, digital annotations can be leveraged to perform actions their analog counterparts are incapable of. For example, the MATE system [64] allows users to turn some specific handwritten marks into actual content edits. The digital world is also particularly effective at indexing content, as demonstrated in XLibris [58] and InkSeine [76]. Both systems infer queries from freeform annotations in order to bring up results that are likely to be of interest to users. XLibris builds queries from the content to which annotations are anchored, while InkSeine directly uses the handwritten words. In both, the system makes the hypothesis that annotations capture users’ interests, and that they can act as starting points for further investigation. Very recently, the ActiveInk system 1 has gone one step further, making it possible to turn annotations made on data visualizations into analytical actions on the underlying data, such as filtering items out.

A frequent problem with annotations on printed paper is the lack of available space to accommodate them. Electronic documents offer opportunities to overcome this limitation. The DIZI system [3] partially addresses this problem by facilitating pen input in small spaces, thanks to a magnifying lens that pops up when users start annotating. Space between lines gets larger, making handwriting with a stylus more comfortable. However, the available space for annotations remains limited, and handwritten text can be very difficult to read when not magnified. Chang *et al.* [31] consider the problem from a navigation perspective. Frequent movements between

2. MAKING SPACE FOR EXTERNALIZING THOUGHTS ON REFLOWABLE DOCUMENTS THAT SUPPORT VARIOUS CONTENT

pages are often poorly supported when reading on a digital device [158], and they introduce an architecture that facilitates navigation between *primary* material and *supporting* material in order to minimize such movements. They describe several techniques such as moving blocks and compressing interlines of primary material, or adding an overlay to make space for supporting material in the context of the primary material. While their system has not been implemented for ink-based annotations but rather for supporting material such as footnotes, it is relevant here as it transiently alters the layout of the primary material to make space for additional, related material.

Existing software implement more radical solutions, such as inserting user interface components floating over the text (*e.g.*, digital Post-It notes) or embedded within the text (*e.g.*, text boxes); or switching to a different view mode displaying a dedicated comment sidebar. However, these techniques interrupt the users' workflow as they require navigating menus and, in some cases, moving and resizing interface components. This disruption may impact the active reading process and interfere with the capture of fleeting thoughts. TextTearing techniques by Yoon *et al.* [193] offer pen+touch interactions that enable a more fluid workflow. Users explicitly create white space boxes between paragraphs to accommodate their annotations, thus actively changing the document's layout. The canvas of the page is resized, altering the *page aspect ratio* across the document, so as to preserve its pagination and ensure that all pieces of content remain on their original page. In their study, Yoon *et al.* found that participants preferred annotating in the extra space created with the technique than in the white space that was there in the original layout. The approach lets users integrate in-context annotations tightly with the document's content, which is particularly efficient when collaborating with other users. RichReview [194] implements TextTearing techniques coupled with the capability to record speech and deictic gestures associated with their annotations to facilitate communication between co-workers.

Our design space includes the above TextTearing techniques, but explores a wider range of possibilities to make space for in-context annotations. It relaxes constraints on document layout much further, taking advantage of content reflowing techniques, that provide opportunities not explored so far. Beyond space-making strategies, it also considers the different moments *when* it might be appropriate to make space for annotations from an interaction perspective, as we detail later.

2.1.3 Overlay Ink and Reflowable Documents

The HCI community has already started studying the problem of embedding annotations into reflowable documents such as Web pages. It has investigated several questions: how to store annotations and access them later on [129]; how to reconcile them with their scope when a Web page's content changes [28]; how to share them with other people [29, 194].

Closer to our work, other projects have looked at the problem of making annotations behave properly in case of content reflowing. This typically occurs when looking at the same page on another device, but on the same device as well, for instance when the browser window gets resized or when the font size gets changed [28, 57].

Systems such as *u-Annotate* [32] and *iAnnotate* [128] let users annotate Web pages with handwritten annotations. Both are implemented as plug-ins that put annotations on a transparent layer on top of the page. Annotations are anchored to the closest

HTML element, so that their position can be properly restored no matter the actual layout rendered on screen.

Golovchinsky *et al.* also investigated the problem of making freeform annotations robust to content reflowing for non-paginated documents in XLibris [57]. The system computes the scope of the ink marks (the words that users have circled or underlined with the pen) so as to not only restore the position of those marks, but properly adapt their shape as well: they can get stretched or split to remain aligned and coherent with their initial scope, which might end up distributed on different lines or even successive pages. Barger and Moscovich [13] studied how users react to such techniques. They found that users like the automatic adaptation to the reflowing when the system does it properly; and that users prefer it when the system beautifies their freeform annotations, as opposed to re-rendering them with their original drawing style. While these projects have focused on making overlay ink reflow with the content, the SpaceInk design space proposed here is about techniques to make room for in-context annotations.

To summarize, previous work explored multiple aspects of digital annotation, suggesting strategies to reflow overlay ink as the document content evolves, as well as devising interaction techniques for making room for annotations within the content. Our research advances our understanding of in-context annotations, providing a space for reflecting on design dimensions that impact the user experience, and reporting on a study shedding light on their implications. Our work is also the first to consider how both overlay marks and in-context annotations coexist during active reading (Figure 2.10) with the goal of identifying a set of considerations about the design of interactions to fluidly interleave them.

2.2 SpaceInk

In this section, we first map the space of interaction techniques for in-context annotations, organizing them around two salient dimensions. We describe our rationale, as well as a set of representative techniques that we implemented for further empirical study.

2.2.1 Design Space

Interaction techniques for in-context annotations consists of space-making actions (causing the document content to reflow) and inking actions (freeform input of the annotation content). SpaceInk organizes them along two dimensions, as illustrated in Figure 2.1:

- *where* the annotation is inserted relative to the corresponding content – annotations can be inserted *between words*, *between lines*, or they can be *wrapped in a paragraph* or *put on the side of a paragraph*;
- *when* do users interact to make space for the annotation – they can push content *before* annotating, *while* annotating, or *after* annotating.

Considering *where* the annotation is inserted is important, as it defines the spatial proximity with its scope – the content to which the annotation is anchored to. Allowing users to precisely adjust their annotations' position ensures that in-context annotations are rendered as close as possible to the content they refer to.

2. MAKING SPACE FOR EXTERNALIZING THOUGHTS ON REFLOWABLE DOCUMENTS THAT SUPPORT VARIOUS CONTENT

Considering *when* the space is made for the annotation is also important, as this interaction may interfere with active reading. Deciding of the location and size of the space needed beforehand takes the focus away from the capture of fleeting thoughts, while doing it after may obfuscate key content that could be necessary to finish formulating an idea.

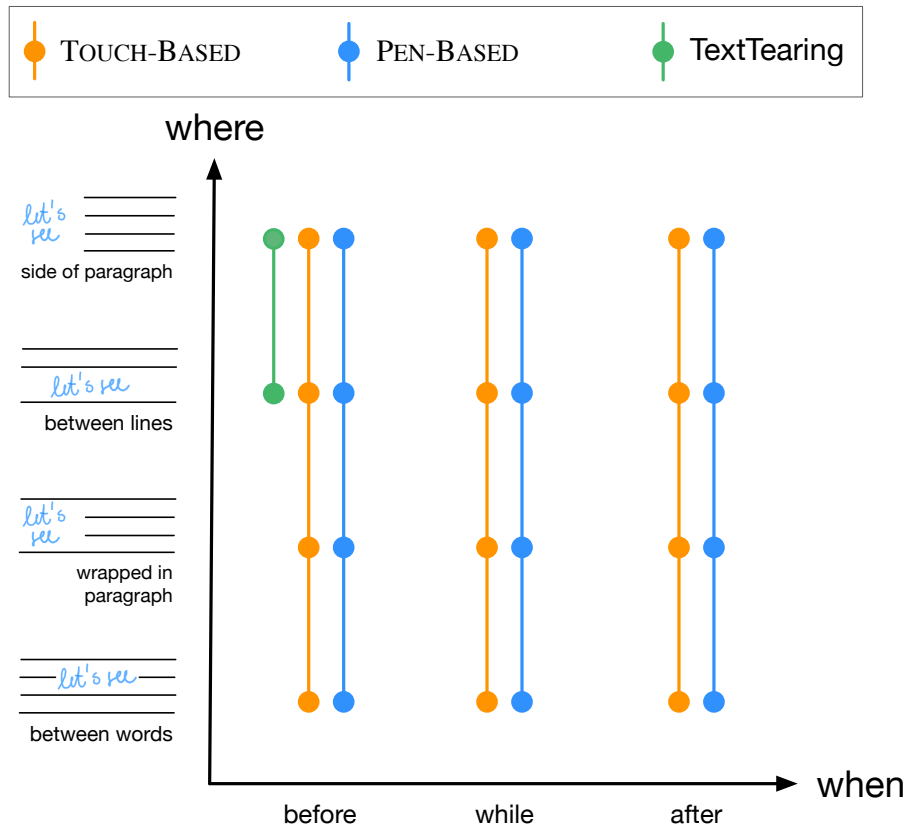


Figure 2.1: SpaceInk is organized along two dimensions: *when* and *where*. It includes the six techniques tested in our study (orange and blue), as well as TextTearing techniques [193] (green).

Figure 2.1 shows where TextTearing techniques [193] are situated in this space. They mostly fall in cell *where* = *between lines* and *when* = *before*, as they require users to first create some space between paragraphs (*i.e.*, between the last line of a paragraph and the first line of the following paragraph), and only then put ink in that space. Yoon *et al.* also describe a *margin* technique that allows users to widen a document’s margin to get more space for annotations. We position this technique in the $\{side\ of\ paragraph \times before\}$ cell, as it corresponds to the case where users create extra space on the side of paragraphs. It is not an exact fit though, as the technique from [193] affects all paragraphs that belong to the page.

This design space opens up new possibilities regarding strategies to make space for in-context annotations, that we start to explore with examples of techniques in the next section.

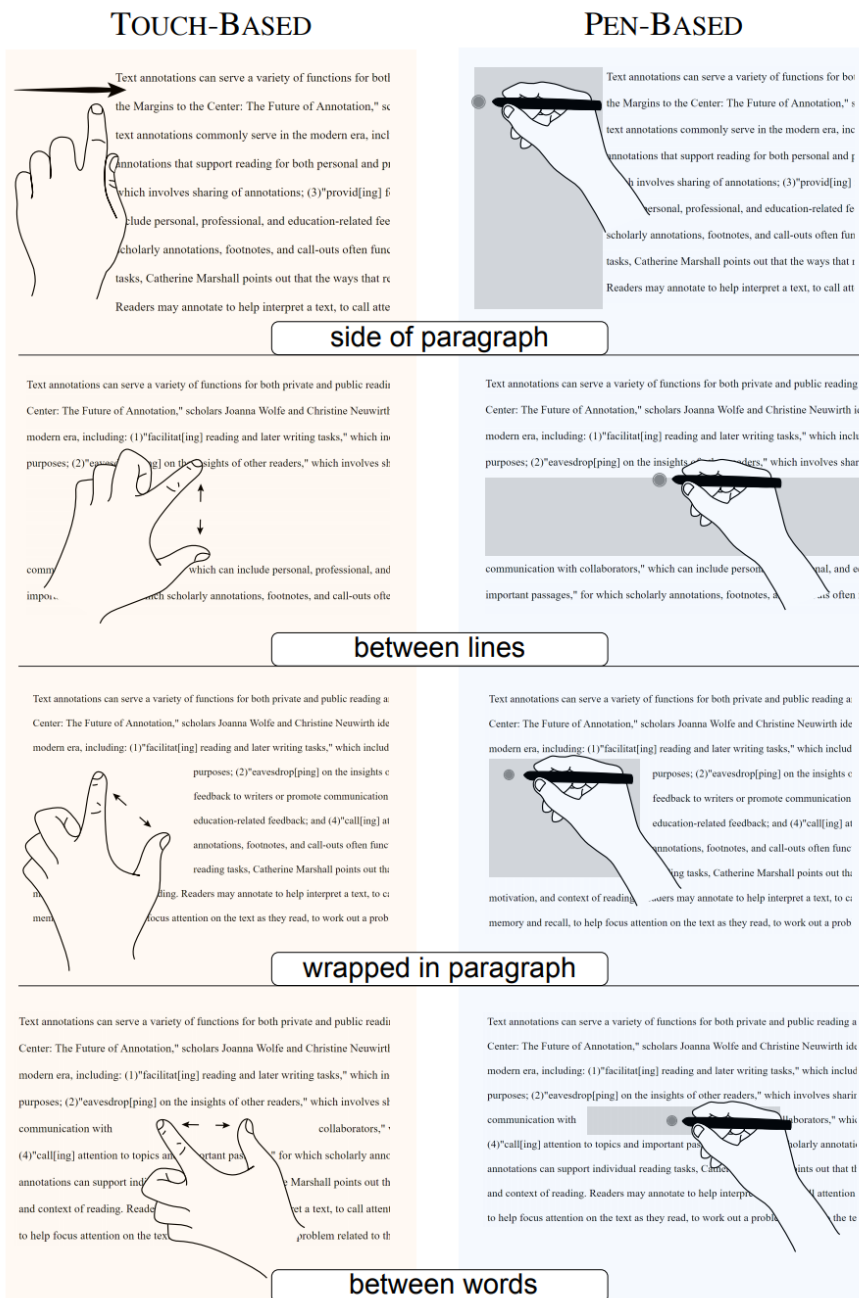


Figure 2.2: Both Touch-Based and Pen-Based techniques let users specify *where* they want annotations to be included relative to the content.

2. MAKING SPACE FOR EXTERNALIZING THOUGHTS ON REFLOWABLE DOCUMENTS THAT SUPPORT VARIOUS CONTENT

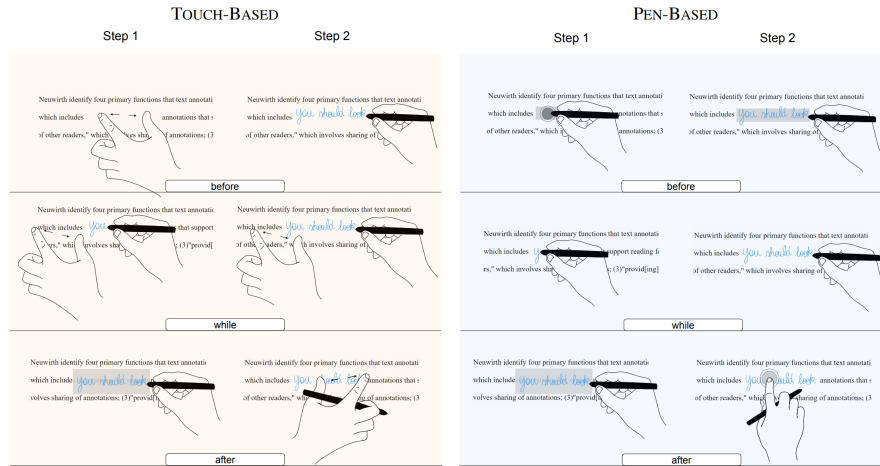


Figure 2.3: Both Touch-Based and Pen-Based techniques let users specify the strategy for creating white space at different moments (*when*): either *before*, *while*, or *after* annotating. With Touch-Based techniques, users were free to both gesture *and* write with a single hand (as illustrated in the *after* condition) or with two hands (as illustrated in the *before* condition).

2.2.2 Implemented Techniques

We used SpaceInk to generate two sets of techniques: those that are Touch-Based, and those that are Pen-Based (Figure 2.2). Each set is comprised of three techniques (the vertical lines in Figure 2.1), one per value of *when*. For each technique, users choose *where* to make room for annotations among four options, as detailed below:

- Touch-Based (Figure 2.2, left column): Given the ubiquitous use of single-finger panning gestures to scroll Web pages while reading them, we decided to co-opt the pinch-to-zoom gesture instead. We reasoned that zoom level would likely be set once before the active reading started and seldom adjusted. Co-opting it thus seemed less disruptive to users' workflow. Diagonal pinch gestures¹ create white space wrapped inside text; vertical gestures create space between lines; and horizontal gestures create space between words. However, as pinch-to-zoom proved often awkward to perform in tight spaces close to the screen bezel, we decide to use a swipe gesture for margin expansion, as proposed in [193]. This gesture can be distinguished from horizontal scrolling in web pages or page flipping in paginated documents by recognizing a swipe starting from the bezel [139].
- Pen-Based (Figure 2.2, right column): different types of white space are generated depending on *where* the pen is when users start annotating. If starting in the margin of a paragraph, the paragraph gets pushed towards the right. If starting

¹whose direction $d \in [20^\circ, 70^\circ]$ in our current implementation.

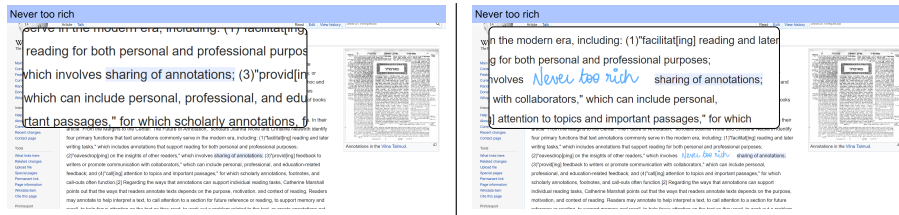


Figure 2.4: Experimental task. (left) stimulus: a participant is instructed to annotate the highlighted text sharing of annotations with words *Never too rich*. (right) response: she annotates in-context, making space for her ink using a *between-words* strategy.

between two lines, the lines below the pen's position get pushed downward. If starting on the first word of a line, an empty box is added there and the paragraph's text gets wrapped around it. Finally, if starting on another word, or between two words of the same line, words that follow the pen's position are pushed toward the right, reflowing all sentences that follow (if need be).

Figure 2.3 illustrates how gestures (or pen-down events) and actual inking actions are temporally organized in each of the six techniques:

- Top row: users create space *before* inking. Touch-Based case: the size of the white-space box is defined by the amplitude of the pinch gesture. Pen-Based case: a box with a predefined size is created when the pen hovers the surface. In the current proof-of-concept implementation, assuming an average font size of 3mm (ascent+descent), a box on the side of a paragraph is 19mm wide; a box wrapped in a paragraph is 19×19mm; a box between lines is 9mm high; a box between words is 38mm wide.
- Middle row: users create space *while* inking. Touch-Based case: white space is created by means of a pinch gesture. Pen-Based case: white space is automatically added as the annotation gets longer, so as to prevent it from overlapping the document's content.
- Bottom row: users directly ink above the text and create white space to accommodate their annotations *only after* they are done inking; using a pinch gesture (Touch-Based case) or tapping on the ink to tell the system to create the necessary amount of white space (Pen-Based case).

We implemented all six techniques and iterated on them through pilot studies.² We eventually decided to discard the Touch-Based technique that lets users adjust white space *while* inking, as it proved too difficult for users to perform a pinch gesture while writing at the same time. We identified properties of the remaining five techniques that might affect user experience. In particular, the extent to which content adaptation is under the user's control is higher for Touch-Based techniques than for Pen-Based techniques. With Touch-Based techniques, the pinch gesture acts as a rubber-band

²Short videos demonstrating each technique are available at <http://ilda.saclay.inria.fr/spaceink>.

interaction that defines the location and size of the created space. With Pen-Based techniques, which rely on a single input channel, the control of the white-space area is merged with the act of annotating. The system plays a more active role in the specification of the area's dimensions, which is either predefined (*before*) or automatically computed by the system (*while/after*).

2.3 User Study

The aim of our experiment is to gather qualitative feedback about these representative techniques, and to relate their performance with the contexts in which annotations are made. Based on the definition of an annotation as a marking made of some *content* and an *anchor* from [28], we introduce the following two factors to operationalize different contexts in which annotations are made:

- annotation *Length*: Short, Long, or Expanding; and
- annotation *Scope*: Inline or Block.

The annotation's *Length* corresponds to how many handwritten letters and symbols it contains. This number can vary with the annotation context. For example, personal annotations tend to be tacit and short, as opposed to annotations to be shared, which are more verbose and explicit [103]. In addition to these two cases (Short and Long), we also consider Expanding annotations in order to operationalize the situation where users are incrementally structuring their thoughts, or where they want to add more ink to an annotation, as a result of further investigation. This contrasts with the study reported in [193] where users were always able to anticipate the content, and thus the length, of their annotation. We believe that this is an important factor to consider, as long or expanding annotations make it difficult or impossible for users to anticipate how much space they will need to fit their markings. We expect this factor to impact *before* techniques particularly. Indeed, users have to anticipate how much space to create so as to make the annotation fit, which induces much premature commitment [61]. This leads to our first hypothesis:

- H_1 : the usability of *before* techniques is affected by the length of the annotation.

The *Scope* factor can take the two values described by Marshall [102]: Block corresponds to her *margin* anchor (*i.e.*, the annotation's scope is a paragraph), and Inline to her *range* anchor (*i.e.*, the annotation's scope is a highlighted portion in a paragraph). As people like to ensure proximity between an annotation and its anchor [103], *Scope* might have an impact on their annotation strategy. This leads to our second hypothesis:

- H_2 : users want to embed the annotation inside the content when the anchor is a portion of a paragraph (Inline).

The hypotheses formulated above suggest that there is no *a priori* clear winner, but rather that each technique might have strengths and weaknesses. Our study aims at identifying what strategies were effective and in what context. We follow a within-subject design where participants have to perform annotation tasks under the above six conditions, with each of the five techniques introduced in the previous section:

Touch-Based+*before*, Touch-Based+*after*, Pen-Based+*before*, Pen-Based+*while*, and Pen-Based+*after*. In all conditions, participants are instructed to put annotations as close as possible to the highlighted anchor, and to minimize the amount of wasted white space.

2.3.1 Task

The experimental task is illustrated in Figure 2.4. Participants are presented with *i*) a document in which a specific text fragment is highlighted (the annotation's anchor), and *ii*) the text of the annotation they have to write. We deliberately designed a task focused on low-level aspects of interaction (visual perception, motor control), avoiding higher-level cognitive processes that would have added noise due to inter-user variability in the handling of different kinds of annotations. The document to annotate was derived from the Wikipedia page about text annotation. The contents of annotations come from MacKenzie and Soukoreff's phrase set [101], and are not semantically related to their scope. Short annotations consist of 3 words; Long ones of 7. In the case of variable-length annotations (random length between 5 and 10 words), the system starts by showing the first 3 words, revealing one more word every two seconds until all words are shown. An ellipsis (...) is shown to indicate when more words are yet to be revealed.

2.3.2 Participants and Apparatus

Twelve unpaid volunteers (4 female), daily computer users, age 23 to 43 year-old (average 27.3, median 25), served in the experiment. It was conducted on a Microsoft Surface Book 2, equipped with a 13" screen (resolution 3000×2000 pixels) that supports multitouch and pen input. Participants were encouraged to take the device and adjust their hold for comfortable pen and touch interaction.

2.3.3 Procedure

Participants first sign a consent form and fill out a demographic questionnaire. The experiment is then split in two blocks, one per set of techniques (Touch-Based and Pen-Based). Each block is then divided into sub-blocks, one for each technique (two for Touch-Based, three for Pen-Based). The presentation order of blocks and sub-blocks is counterbalanced across participants. In each sub-block, the operator briefly introduces the technique using a short video clip, and then lets participants train with the technique.³ After this training phase, participants complete a series of 12 trials, 2 replications presented in a row for each *Length*×*Scope* condition. The presentation order of these conditions is counterbalanced across participants as well. At the end of each sub-block, participants are asked to rate the technique along five 6-point Likert scales for: physical comfort, cognitive load, enjoyment, efficiency (for Short, Long and Expanding annotations separately), and system predictability. At the end of the experiment, participants are further asked to rank the five techniques along these same axes (with the exception of system predictability, which might sound too abstract to participants). We use this final ranking in order to sanity-check that individual scores given after using each technique actually reflect participants' relative ranking of techniques (as individual scores might have been influenced by the presentation order).

³This training phase consistently lasted about 5 minutes for all participants, even though they were not given any time limit.

2. MAKING SPACE FOR EXTERNALIZING THOUGHTS ON REFLOWABLE DOCUMENTS THAT SUPPORT VARIOUS CONTENT

Participants are encouraged to verbally share their impressions all along the experiment. In addition to audio recordings, the operator writes down a summary of each participant's feedback. The system logs participants' strategy for annotating (*i.e.*, where they create space), and takes screenshots at the end of each trial. The whole procedure lasts around 75 minutes.

2.3.3.1 Data Collection & Data Analysis

Our experimental software collected data about how users made space for their annotations, where they made space in the document, the amount of time they took to perform each gesture. It also recorded the final set of annotations before proceeding to the next trial. We also audio recorded participants to capture their thoughts and comments. A single experimenter was present for each session. The experimenter retrieved comments and feedback using pen and paper notebook.

We then analyzed all data gathered depending on *where* people have annotated. We categorized feedback and comments as follows: positive feedback, negative feedback, problems encountered with the interface and wishes about additional functionality. Finally, we compared feedback from participants and results from the study to better understand people strategies' to make space for annotations.

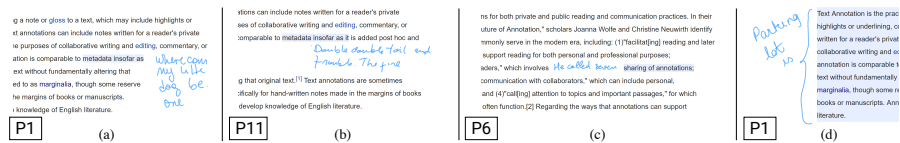


Figure 2.5: Sample participant annotations illustrating different strategies.

2.3.4 Results

Figure 2.6 shows what strategies (*where*) participants used to annotate and Figure 2.5 illustrates examples resulting from these different strategies. Data and screenshots collected during the study, as well as additional charts, are available at <http://ilda.saclay.inria.fr/spaceink>

All participants experience each technique. The experiment design can be summarized as follows:

	12	users
×	5	techniques
×	3	levels of annotation <i>Length</i>
×	2	levels of <i>Scope</i>
<hr/>		
=	360	trials in total

2.3.4.1 People use the full range of strategies

Overall, participants mostly wrapped their annotations in paragraphs (36.4%) or inserted them between lines (34.2%) (Figure 2.6-a). Strategies that consist of annotating between words (16.9%) or on the side of a paragraph (12.6%) were used less frequently. They favored the three strategies that embed annotations inside the content over

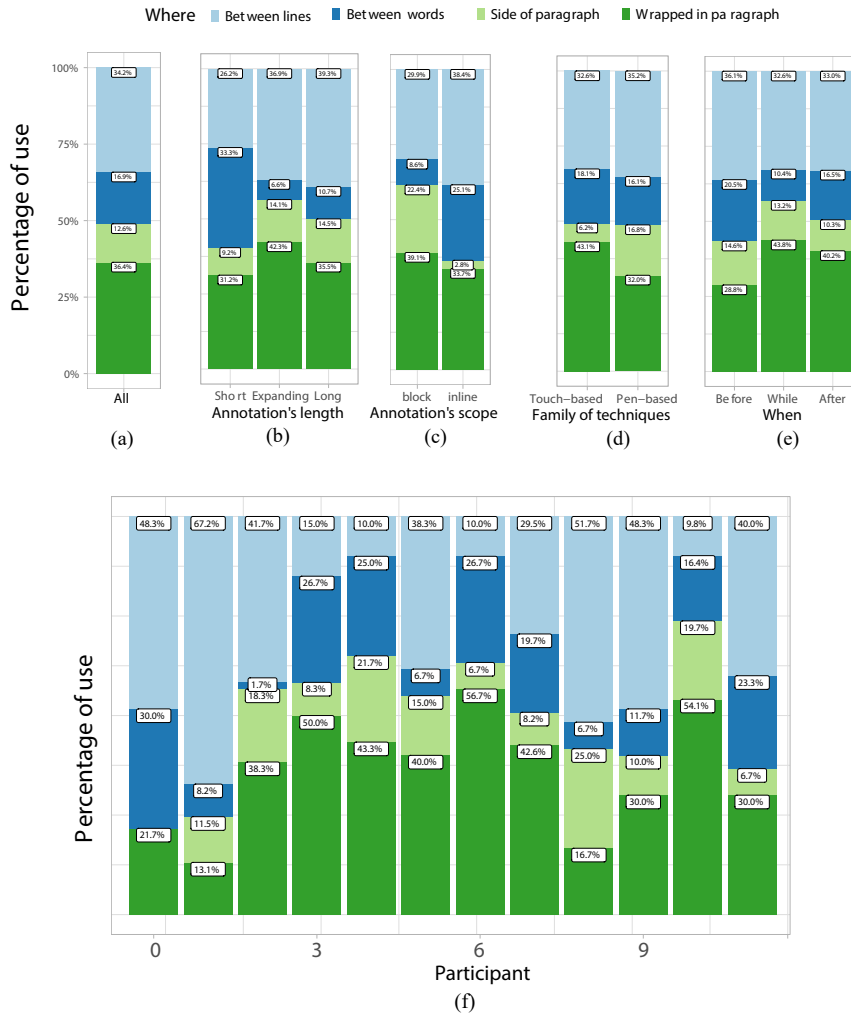


Figure 2.6: Distribution of participants' space-making strategies (*where*): (a) for all trials, (b) per *Length*, (c) per *Scope*, (d-e) per technique conditions, and (f) per participant.

annotations on the paragraph's side. This is consistent with findings reported in [193]. A breakdown of the distribution of strategies per annotation *Length* (Figure 2.6-b) also reveals that participants tend to embed annotations between words when they are Short (33.3%), but not when they are Expanding (6.6%) or Long (10.7%). This effect is not very surprising, as our implementation of the *between-words* strategy is limited to a single line. As soon as the whitespace area is taller than a line, it becomes wrapped in paragraph.

What is particularly interesting to contrast with previous findings is that participants did adopt the *between-words* strategy. They used it to closely integrate annotation and content, especially to annotate a portion of text within a paragraph (Inline). Figure 2.6-c shows that the *between-words* strategy was often chosen (25.1%) in the Inline condition.

2. MAKING SPACE FOR EXTERNALIZING THOUGHTS ON REFLOWABLE DOCUMENTS THAT SUPPORT VARIOUS CONTENT

Comparing the two *Scope* conditions, we can see a clear inversion in the distribution between *side-of-paragraph* and *between-words* strategies. This is in line with our expectations about users' will to minimize the distance between an annotation and its anchor (H_2).

The technique itself does not seem to influence the choice of strategy (Figure 2.6-(d-e)). What is noteworthy, however, is the variability between participants (Figure 2.6-f). During the experiment, the operator noticed that participants had their personal preferences regarding their strategy, which remained rather consistent all along the experiment. Figure 2.6-f reports the distribution of strategies per participant, and Figure 2.5 shows sample trials that illustrate those different strategies. For instance, participant P11 liked using vertical expansion. She used it both when the anchor was the Block and when the anchor was Inline, by creating some space below the paragraph or below the line containing the highlighted portion of text. In contrast, participant P6 wrapped almost all his annotations in the text, whatever the anchor, without worrying about consequences in terms of content reflowing. Finally, although the annotations' anchor was always highlighted, two participants chose to visually represent their scope (see P1's bracket in Figure 2.5-d). These observations support the fact that the *strategy for annotating is personal*, and that techniques should offer as much flexibility as possible to accommodate a wide variety of users.

2.3.4.2 People do not estimate required space accurately

The screenshots taken by the system at the end of each trial reveal that participants were often unable to estimate the length of their handwritten annotations, and ended up annotating over the text with *before* techniques in some cases. This is in line with hypothesis H_1 but, surprisingly, this happened even in the case of Short annotations (Figure 2.7). The cost of premature commitment and its impact on the perceived usability of *before* techniques is reflected in participants' ratings. They found *before* techniques much less efficient than *while* and *after* techniques. They also often complained about the predefined size of boxes with the Pen-Based+*before* technique.

Participants' ratings, reported in Figure 2.8, suggest that Touch-Based techniques were rated slightly higher regarding predictability, as users explicitly define the space for annotating with a pinch gesture. But Pen-Based techniques were enjoyed much more. Participants found them more comfortable, and easier to use. This suggests a trade-off between predictability and fluidity: while Touch-Based techniques are more predictable, users might favor the fluidity offered by Pen-Based techniques, which rely on a single input channel for completing all steps in the annotation process. Interestingly, the difference in terms of predictability is quite small. Although there were some cases where participants got surprised by the space-making strategy the system applied when using Pen-Based techniques – especially so with the *after* version – participants quickly got accustomed to the fact that the choice of strategy was driven by the location of the first pen-down event. We even observed that 10 out of 12 participants tapped on the predefined area before actually annotating when using the Pen-Based+*before* technique. The initial tap was a means to explicitly tell the system where to make room for the annotation.

2.3. User Study

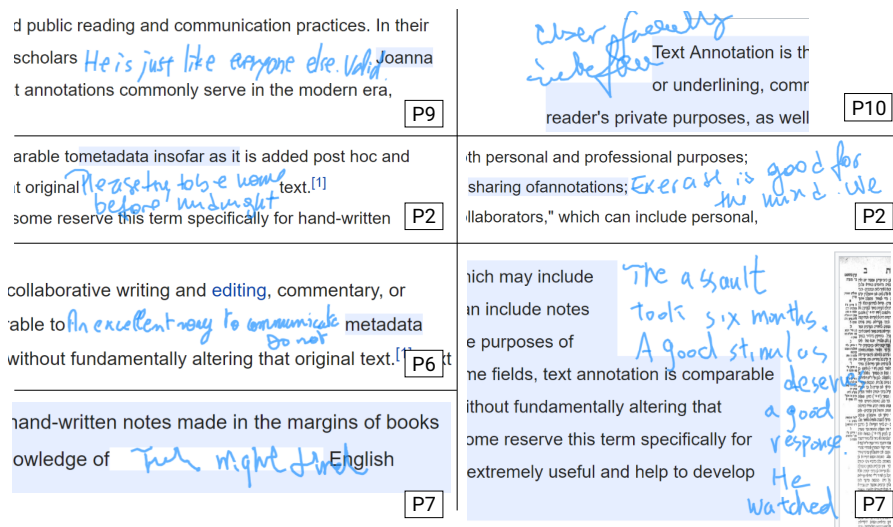


Figure 2.7: Sample trials with techniques making space *before* inking, in which participants had difficulties anticipating the length of annotations.

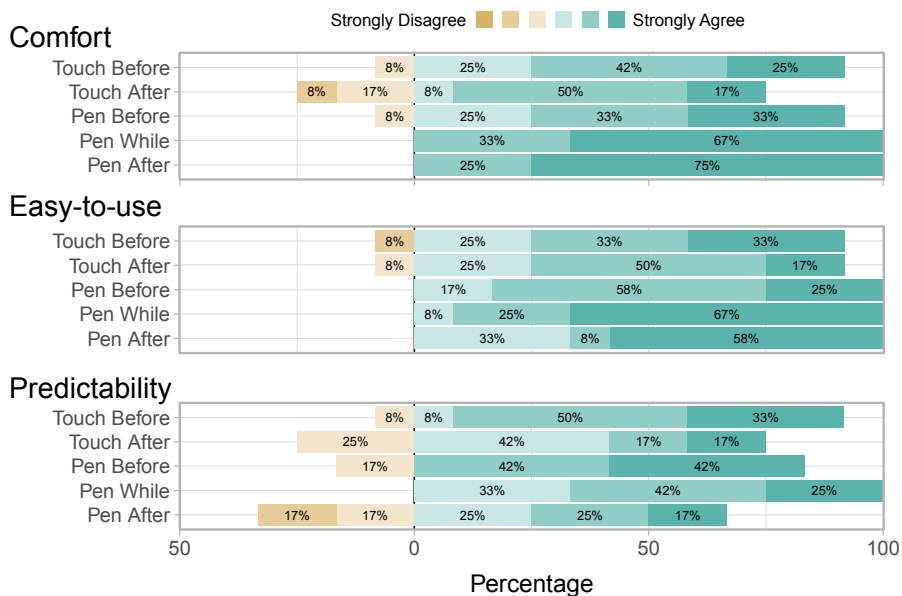


Figure 2.8: Participants' ratings of the five techniques.

2.3.4.3 Less agency may prove more comfortable

A few participants commented that they did not like writing over the text when using *after* techniques. However, participants' average ratings about comfort do not strongly reflect this: both Pen-Based+*while* and Pen-Based+*after* received positive scores. Participants actually rated these two techniques high, as they really enjoyed not having to perform gestures. Interestingly, the fact that the text was dynamically reflowing

as they were inking (in the Pen-Based+*while* condition) did not seem to disturb them. This suggests a trade-off between visual interference and the number of actions: users might find it somewhat disturbing to annotate on top of content, or to see the document getting continuously reflowed as they write, but they find this preferable to performing more interaction steps. However, this observation might not hold in more realistic contexts where users check some elements downstream in the text while annotating. In such cases, the reflowing of content might be a nuisance.

Pen-Based+*while* and Pen-Based+*after* seem to be the favored techniques overall, but results remain contrasted. For instance, in terms of predictability, Touch-Based+*after* fares better than Pen-Based+*after*. This is reflected in participants' comments that praise the quality of all techniques. Five participants actually commented that they would like to have a tool that integrates the best of each approach. For example, P3 said: "*I could clearly see where each technique performed well. If you merge them into one, I would use such a tool.*"

2.4 In-Context Annotations In Practice

The above user study sheds light on different aspects of in-context annotations, that require interleaving space-making interactions and inking. So far, we investigated two techniques in each cell of the design space (pen-only and pen+touch), favoring easy-to-learn and -use techniques based on common gestures. Our main goal was to understand what strategies were effective and in what context. But our general intent is to better support users over the entire document annotation process: when they are making in-context annotations, but when they are inking overlay marks as well. Aiming at streamlining these different types of annotations and considering insights from our study, we identify three key design requirements:

- R_1 : enable lightweight, seamless transitions between overlay ink and in-context annotations, as users make use of both, simultaneously;
- R_2 : support several space-making strategies for in-context annotations, as the choice of strategy is both context- and user-dependent;
- R_3 : make overlay ink reflow with content, as making space for in-context annotations causes such content reflowing.

We describe our rationale for a Web-based annotation environment that meets these requirements. Implemented in Javascript as a Google Chrome extension, this proof-of-concept allows us to test SpaceInk techniques with various devices, annotating arbitrary single-column HTML pages as well as fixed page size formats after conversion (for instance, PDF documents), as demonstrated in the companion video.

2.4.1 Seamless Transitions between Inks (R_1)

During active reading, people interleave marks such as circles or underlines to highlight some portion of content to explicitly *anchor* their thoughts, which they express through the *body* of their annotation [104]. Letting users make both overlay ink marks *and* space for handwritten in-context annotations is thus essential. It alleviates constraints imposed by the document's layout and lets them focus on capturing their thoughts.

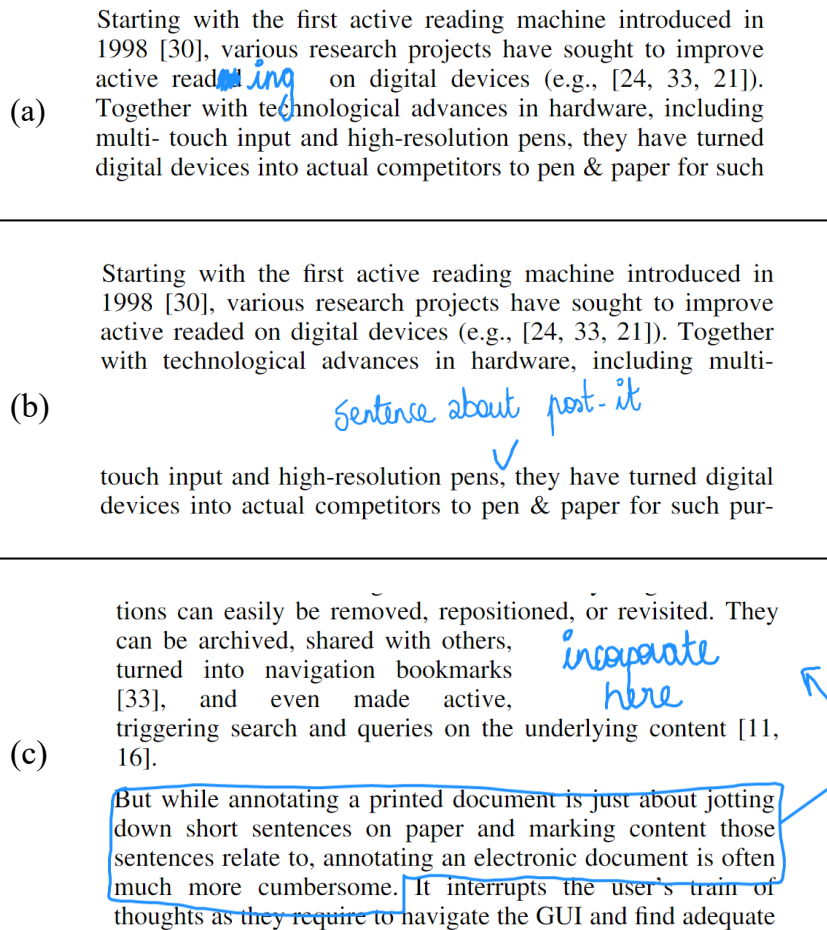


Figure 2.9: Using both overlay ink and in-context annotations to fix a typo (a), or to suggest higher-level revisions (b-c).

Figure 2.9 illustrates such simple cases. However, we must be careful about the cost of mode-switching between the two types of ink.

The mode-switch problem is ubiquitous in UI design, and HCI researchers have already faced it in similar contexts, for instance when designing systems where freeform ink coexists with ink-based commands (or stroke shortcuts [7]). The Inferred-Mode protocol [147] addresses the problem by attempting to infer users' intent from the context. By doing so, it removes the need for an explicit delimiter to switch between modes. This is an elegant solution, but because of the high variability in handwriting and annotating, it is unclear in our context which criteria could effectively disambiguate between the two types of inks. The very same variability makes pigtail delimiters [73], as used in TextTearing [193, 194], prone to false activations. For instance, a curly bracket might get confused with a pigtail gesture.

2. MAKING SPACE FOR EXTERNALIZING THOUGHTS ON REFLOWABLE DOCUMENTS THAT SUPPORT VARIOUS CONTENT


We rather opted for prefix flicks [195]: by default, the pen overlays ink on top of the content, without changing the layout; but when performing a flick gesture, the pen turns into an in-context annotation tool. This delimiter is particularly interesting because it allows users to specify the space-making strategy while switching mode: the strategy is inferred from the relative position of the initial point of contact and direction of the flick gesture, as detailed next.

When the flick gesture is initiated in a margin, the subsequent annotation will push the paragraph, widening that margin. When performed over text, i) a diagonal flick gesture sets the strategy to *wrapped in text*; ii) a horizontal one sets it to *between words*; and iii) a vertical one sets it to *between lines*. Pen-based flick gestures avoid breaking

←
side
of
paragraph

↓
between lines

the fluidity of interaction, as they only involve one input modality and avoid artificial pauses. The pen automatically reverts back to overlay-ink mode when its location is more than 1cm from the bounding box of the in-context annotation. This solution avoids both bezel swipe gestures, which are usually dedicated to existing commands [139], and pinch gestures, which are often already assigned to navigation actions [141]. An alternative to prefix flicks for mode switching could come from [30]. For example, users could slide their index down the barrel of the pen to touch the surface with their finger while inking with the pen, which could be discriminated from pen-only input.

In addition to the above, a SpaceInk icon , displayed in the bottom left corner, slowly pulses when SpaceInk is activated. Users can rely on this explicit representation to check the current mode, and can tap the icon to force a mode switch. They can also hold their finger still on the icon, using it as a spring-loaded control [75] to change the current mode temporarily.

2.4. In-Context Annotations In Practice

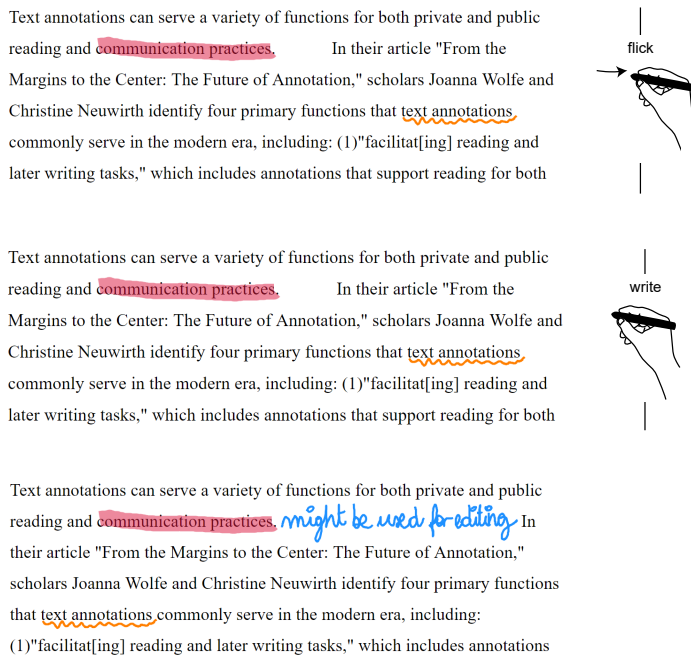


Figure 2.10: In-context annotations coexist with overlay ink. (a) A user highlights and underlines some content (overlaid ink). (b) She performs a horizontal flick gesture to add some white space between two words. (c) She inserts an in-context annotation. White space automatically expands to accommodate the annotation, causing the content to reflow. All ink marks overlaid on content remain spatially-aligned with their scope.

2.4.2 Combining Strategies for Making Space (R_2)

Observations from our study can inform the design of input techniques that make space for in-context annotations:

- G_1 : rely, as far as possible, on pen input only;
- G_2 : avoid premature commitment;
- G_3 : enable users to easily switch between different space-making strategies.

We address G_1 and G_2 by combining prefix flicks to specify the space-making strategy (described earlier) with techniques that adjust the white-space area *while* inking (Figure 2.3, middle row). The created white space incrementally expands to accommodate the annotation, as the user writes it. We also enable users to make more space on demand. Selecting an annotation activates it, enabling users to adjust the underlying white space using a pinch gesture, or manipulating the handles of its resizing box. That box is transient, progressively fading out when not interacted with, and instantaneously disappearing as soon as users start annotating again. Giving users control over the white space area's dimensions is important. For instance, it lets them prevent dynamic content from reflowing if they would rather not have it move while they are annotating. It also lets them create some white space for, *e.g.*, sketching, and

2. MAKING SPACE FOR EXTERNALIZING THOUGHTS ON REFLOWABLE DOCUMENTS THAT SUPPORT VARIOUS CONTENT

Text annotations can serve a variety of functions for both private and public reading and communication practices. In their article "From the Margins to the Center: The Future of Annotation," scholars Joanna Wolfe and Christine Neuwirth identify four primary functions that text annotations commonly serve in


check
refs

```
<body>
  Text annotations can serve
  a variety of functions for
  <span
    class= "floatSpace"
    id= "floatElement"
    style= "
      width: 148.636px;
      height: 180.909px;
      float: left;" >
  </span>
  both private and public
  reading and communication
  practices. In their article
  ...
</body>
```

Figure 2.11: An annotation's spatial scope is encoded as a `` element in the DOM of the Web page.

then freely draw strokes anywhere in that space, since those strokes do not have to remain inside the 1cm-distance threshold introduced earlier.

We address G_3 by adding *Touch-Based-after* to the suite of techniques available in the environment. At any moment, users can grab an existing in-context annotation with a finger and move it. The associated space-making strategy gets automatically updated based on the finger's location (in the margin, between lines, on the first word of a line, anywhere else on a line). When the strategy is updated, white space around the annotation gets cropped, minimizing wasted space. This also provides users with a way to quickly optimize space *a posteriori* when the extra white space is deemed unnecessary.

In addition to addressing the above guidelines, the system lets users turn in-context annotations into overlaid ink and *vice versa*. Annotation bounding boxes (which appear when tapping on them) are decorated with the SpaceInk icon . Tapping that icon toggles between states (overlaid or in-context).

Finally, users can restore the document's original layout at any time by performing a hand swipe (*i.e.*, a swipe gesture using three or more fingers). The document's outer margins grow wider, and all in-context annotations get moved there, the document's content reflowing back to its original layout. Performing the same gesture again puts all annotations back in context, with all user-created white-space areas restored.

In summary, our prototype is designed to support a workflow in which users fluidly interweave both overlay marks and in-context annotations. As shown in Figure 2.10, users only require a pen, which they use both to ink and to make room for annotations. The latter action is triggered by flick gestures, the system automatically creating some white space and expanding it according to the specific gesture made. Users can also explicitly take control on-demand and adjust space precisely, either *before* or *after* the annotation has been written.

2.4.3 Aligning Ink(s) and Content (R_3)

As already mentioned, prior work has studied solutions to keep overlay marks consistent with their scope when the document gets reflowed [13, 57]. In our case, this is of primary importance as in-context annotations can themselves cause the content to reflow. Figure 2.10 illustrates this on a simple example.

To ensure that in-context annotations preserve their position relative to the content, and that overlay ink remains spatially-aligned with its scope, our proof-of-concept implementation relies on a dual-layer UI canvas. Interaction with, and rendering of, annotations are handled in a transparent layer on top of the document using Paper.js (<http://paperjs.org>). White space for annotations is then inserted in the content by adding `` elements of the appropriate size to the HTML page's DOM (Figure 2.11). These `` element are inserted as children of the closest-ancestor block (`<div>`, `<p>`, *etc.*), right after the word that is closest to the annotation's starting point.

Dummy `` elements also get inserted for overlay ink. Their size is null, as their purpose is not to push content, but rather to act as spatial anchors for the overlaid ink marks. As the `` nodes get reflowed along with all other inline CSS elements in the block, ensuring that overlay marks remain consistently aligned with their scope is straightforward. It can be achieved just by listening for DOM reflow events and requesting the upper layer to re-render annotations in the right place. This approach works in most cases, and could be extended with techniques described in [13, 57] to handle more advanced cases where overlay marks should be stretched or split.

2.5 Future Work

One limitation of our approach lies in the technique for positioning annotations relative to the document's content: it assumes that this content does not change. Reconciling annotations with content that does change, as is often the case with Web pages, can be a difficult problem [128]. Additional studies in the spirit of the one by Brush *et al.* [28] are needed to design more elaborate repositioning strategies. Such strategies could prevent, *e.g.*, the orphaning of annotations, which is perceived as an important issue by users [29].

Another question worth further investigation is the potential adverse impact of content reflowing on users' mental map of the document. When performing active reading on paper, people sometimes use annotations to build a spatial representation of the document [121]. If reading the document in a non-linear way, in-context annotations might interfere with this mental process, as they could be changing the document's layout frequently. At the same time, people are increasingly used to switching between different devices that render documents differently depending on factors such as, *e.g.*, screen size. It is thus actually unclear whether they build such spatial maps with digital content. Nevertheless, studying this type of annotations in higher-level tasks than the ones we considered in our study would help understand what are the benefits and drawbacks of content reflowing in terms of cognition.

Finally, we believe that an approach complementary to the one explored in SpaceInk is also worth investigating. Elaborating upon what was initiated with overlay ink in [13, 57], in-context annotations could be transformed dynamically to make them fit inside the available white space while optimizing spatial proximity with their

2. MAKING SPACE FOR EXTERNALIZING THOUGHTS ON REFLOWABLE DOCUMENTS THAT SUPPORT VARIOUS CONTENT

scope. Techniques could include simple affine transforms applied to vector-based ink, morphing annotations (treated as textures) into arbitrary shapes, and reflowing handwritten text, for instance turning a long line into several shorter ones.

PART II

Presenting results of sense-making activities through an expressive design approach

2. MAKING SPACE FOR EXTERNALIZING THOUGHTS ON REFLOWABLE DOCUMENTS THAT SUPPORT VARIOUS CONTENT

Projects in the previous part focused on interactive systems and interaction techniques to support sense-making activities. This second part investigates techniques to visually represent insights and answers to questions gained from such sense-making activities. We focus on one of the most elaborate data structure: *multivariate networks*. Their complex structure can be represented in various manners, using node-link diagrams or adjacency matrices. Visual representations of those structures help users in a variety of tasks, such as understanding relationships between people, visualizing co-occurrences between words, *etc.* When creating a visualization of such structures for communication purposes, *node-link diagrams* remain the favored representation. But the authoring of node-link diagrams from multivariate networks remains a difficult task, especially for designers who want to have flexibility and be able to work iteratively in the design process.

Projects in this part investigate solutions to better support creativity in the design workflow of node-link diagrams, and to extend the set of encoding variables available to designers to represent data attributes associated with the edges of multi-variate networks, by considering motion variables and evaluating their potential from a perception perspective.

Creating expressive node-link diagrams using Graphies

The work presented here is under review as a journal submission. During this project I was responsible for the design and implementation of prototypes and techniques, and I designed and conducted the studies. The paper was written collaboratively by all authors.

The design of a visualization is primarily driven by decisions about chart type and visual mappings, which focus essentially on perceptual effectiveness. But when the primary purpose of a visualization is to communicate a message and engage its audience [145], designers will adopt a more creative process. This process will include considerations about, *e.g.*, memorability and aesthetics [90], which will uniquely influence the decisions made by each designer.

Expressive design approaches bring flexibility in this creative process, enabling designers to easily customize their visualizations and quickly test alternatives. They achieve this in part by removing the artificial boundary between the specification of the visualization on one side, and its interactive customization on the other side. They either integrate both activities in a single environment [90, 97, 145] or build bridges between them [24]. These tools support a wide range of charts, as long as they can be derived from tabular data.

They offer little or no support for graph data, however. Designers who want to create expressive network visualizations must resort to dedicated visualization and analysis software and then customize the output in a general-purpose vector graphics editor. Such software [18, 8, 120] are good *generative tools* [24], but they fail to support the flexible, iterative design process that is key to *expressive* visualization authoring.

Informed by a user-centered design process involving data analysts working with networks, we contribute Graphies,¹ an expressive design environment for creating node-link diagrams for communication purposes.

¹<https://hugoromat.github.io/graphies>



Figure 3.1: Overview of Graphies' Web-based interface, running on a tablet with support for pen+touch.

Graphies, illustrated in Figure 4.1, lets designers incrementally populate a canvas with nodes and links of interest from multivariate graph datasets; specify visual mappings and apply them to arbitrary selections of such elements; and customize the visual appearance of individual elements as well. All such actions can be freely intertwined, providing designers with a flexible workflow that streamlines the creative process [23, 97], effectively supporting quick design iterations. Individual creativity is further supported by the possibility to, *e.g.*, sketch node shapes or interactively bundle or fan links. Graphies also provides designers with communication-oriented features, including an automatic legend generator and tools to export sequences of snapshots or animated movies that convey different perspectives on the same graph or illustrate its evolution over time.

After giving an overview of related work, we describe our user-centered design process. We then discuss Graphies' interaction model and features. Finally, we report on a first-use study in which participants successfully reproduced several expressive designs, and created their own designs as well.

3.1 Related Work

The literature about graph visualization is rich. We limit our overview to the most related systems for general-purpose expressive visualization design on one side, and for visual graph analysis and graph visualization authoring on the other side, as we draw from both research areas.

3.1.1 General-purpose Expressive Visualization Design

In their taxonomy, Grammel *et al.* [59] identify the *template editor* and the *visual builder* as the best user interface scheme for creating visualizations. By using templates as starting points, editors such as Tableau [167] or ManyEyes [178] allow users to quickly design visuals. However, template-based approaches offer limited support

for customization [145] in comparison to visual builders. With the latter, users have more control over how the graphical marks are assembled, and how data attributes are mapped to visual properties.

Mendez *et al.* [111] compare Tableau’s classic *top-down* approach with iVolver’s [112] *bottom-up* (or *constructive* [82]) approach. The top-down approach of Tableau is observed to be faster, but less transparent and less flexible, thus less adapted to the design of personalized, creative visuals. On the contrary, the bottom-up approach of iVolver is observed to encourage users to explore alternative visual designs.

Recent work has focused on environments that explicitly support *expressive* design. Prominent examples include Lyra [145] and Data Illustrator [97]. Users associate data with initially-agnostic vector glyphs, thus avoiding too much premature commitment [61] in the creative process. The two differ in their interaction model for mapping data to visual variables. Lyra works with drag-and-drop actions from data to interactive placeholders on glyphs, while Data Illustrator integrates data bindings as possible values for glyph (or layout) properties, that users can select in inspectors. Charticator [132] follows the same general approach but considers layout as a driving element in the authoring process. With Data-driven guides [90], designers create freeform illustrations in a vector graphics editing environment, and can bind geometric properties of those shapes (length, position and area) to data attributes.

These *expressive authoring tools* offer much more flexibility than *generative tools* (e.g., programming frameworks such as D3 [25], or business intelligence products such as Tableau [167]), the latter often requiring designers to resort to vector graphics drawing tools to manually customize their output [24]. However, while expressive tools enable the creation of a wide range of visualizations, they work with tabular data, and have little or no support for graph data structures. Those that do, such as iVisDesigner [131] or Charticator [132], are forced to expose the concepts of nodes and links in an indirect, somewhat cumbersome manner, failing to explicitly represent the multivariate data in the context of the graph’s topology.

3.1.2 Authoring Graph Visualizations

While the above environments make it possible to create a wide range of visualizations, other systems are specifically designed for working with graph data structures and the specific challenges that they raise. Indeed, representing the topological structure and the attributes associated with the nodes and links of multivariate graphs is often difficult. For instance, spatial position is typically under the control of the graph layout algorithm, which tries to produce a representation that is legible, based on heuristics such as minimizing the amount of link crossings [183]. As a consequence, node size variations are fairly constrained, eventually limiting the number of encoding channels for the visual mapping of data attributes [135]. The problem gets more difficult when graphs grow in size and complexity [179] and when they evolve [89].

Some systems operate at a very low-level: GraphViz [48] consists of command-line programs taking graph data files as input, that users can edit in any text editor to change the appearance of nodes and links. GUESS [2] provides users with an interactive interpreter in which they can perform selections and filtering operations on the loaded data, and specify visual mappings by means of code instructions. D3 [25] provides developers with a full-featured API to generate interactive graph visualizations on the Web. Still on the Web, GraphCoiffure [165] is a system that lets users apply style

sheets to static graphs. These systems have much expressive power but they all require programming skills. Users cannot edit the visual representation directly, which severely limits the accessibility of those tools [90, 145] and does not favor an iterative design process.

NodeXL [164, 162], as a Microsoft Excel plug-in for the creation of node-link diagrams, is accessible to a broader audience. NodeXL lets users specify mappings from data attributes to several visual variables. Users benefit from their experience with the well-known spreadsheet program, but NodeXL has to comply with Excel's model for producing charts: users have to work conjointly with raw data in a tabular form and with the graphics. Again, direct editing of the visual representation is limited.

Focused on data-driven storytelling, DataToon [89] has some commonalities with Graphies, which we discuss in Section 3.3. Its focus is different, however, as it is designed for creating comics-inspired, paginated representations of relatively small dynamic networks [10]; whereas Graphies rather puts emphasis on the creation of elaborate visual mappings for larger multivariate networks.

3.1.3 Visual Exploration of Multivariate Graphs

Beyond the above systems, that are focused on authoring graph visualizations, there is a variety of tools for the visual analysis of graphs. We focus here on systems that provide effective support for the visualization of multivariate graphs. One possibility consists in forcing specific layouts by mapping up to two node attributes to spatial position. Both Semantic substrates [161] and PivotGraph [184] lay out nodes in a scatterplot-like manner. This makes it possible to encode attributes with position, but can also adversely impact the graphs' legibility as node placement is no longer optimized with respect to the earlier-mentioned legibility heuristics. This calls for some interaction in order to enable users to switch between different projected views, an idea that was further explored in GraphDice [22].

The Network Lens [86], as all systems that adopt a focus+context strategy, heavily relies on interaction. Users move a lens on top of the graph represented as a basic node-link diagram to reveal plots associated with its nodes' attributes. Another option consists of aggregating nodes using different clustering strategies based not only on topology, but on attribute values as well [1, 125, 157].

GraphTrail [46] takes a radically different approach, focusing on the multi-variate data associated with the graph's elements, that get displayed using charts that bin nodes and links according to selected attributes. Users can get an overview of the graph by juxtaposing several such charts. The idea of hiding the topology to the benefit of attribute-based statistical charts had already been explored in NetLens [87], but was limited to a specific meta-model at the time. DOSA [176] explores a compromise between this approach and node-link diagrams, embedding the charts into the node-link diagram to represent aggregated nodes.

Systems such as Gephi [18], Tulip [8] and Pajek [120] are designed to support the exploration of large graphs. They typically have a steep learning curve, as they provide users with a rich set of features: filtering, aggregation and analysis functions, as well as presentation functions. Users can change the appearance of graph elements by specifying basic visual mappings. However, these apply globally, limiting flexibility as elements cannot be edited individually by direct manipulation. The workflow is

primarily intended to support data analysis, and the associated interaction model is often indirect and complex [22, 71]. Again, the flexibility and accessibility of the design workflow is limited.

In summary: on one hand, general-purpose expressive visualization environments are accessible solutions that bring much flexibility to the design workflow, but that are not well-adapted to graph data; and on the other hand, graph visualization software are well adapted to these particular data structures, but feature very limited support for expressive design. Our goal with Graphies is to reconcile both, so as to ease the expressive design of node-link diagrams from graph-structured data.

3.2 Design process

We have been working on the design of Graphies over the course of one year with data analysts from a consulting company. The company provides services such as making cartographies of innovation in various domains of activity, by surveying and monitoring the scientific and technological production of the different actors in these domains. The analysts' work is supported by a large database that contains multiple types of resources (actors, scientific articles, patents, technical documents) and multiple types of relations (authorship, collaboration, citation). Based on these data, which form a large, multivariate, and heterogeneous network [119], analysts can picture a complex domain, and advise their clients about opportunities for innovation. Depending on their clients' needs, they work on tasks such as, *e.g.*, identifying the main actors in a given domain, how different domains are articulated, what domains are active in academia and industry. Essentially, they have to find insights in the data, and present those findings to their clients, which often involves creating node-link diagrams.

The first author working in close collaboration with the above-mentioned company, we had the opportunity to involve data analysts regularly in the iterative development of Graphies. We conducted a longitudinal observation that helped us understand their tasks and workflow, the tools they use, how they use them, and what sort of problems they face. These observations informed the initial design of Graphies. We then conducted five sessions of demonstrations and interviews regularly distributed throughout the design process. During those sessions, data analysts played with the latest iteration on the Graphies prototype. We gathered feedback about 1) the features that we had already implemented, 2) the features that were under development, and 3) the features that data analysts wanted us to include.

Our first high-level observation is consistent with what Spritzer *et al.* report about journalists [165]. The typical network visualization construction workflow of these expert users involves at least two types of tools: 1) a graph analysis tool such as Pajek or Gephi to delimit the subgraph of interest and apply coarse-grained visual mappings to it (such as setting node size based on a data attribute, color on another, and showing node labels); and 2) a vector drawing editor such as Adobe Illustrator or even Microsoft Powerpoint to further edit the diagram: making manual adjustments to the layout and visual variables of some nodes and links to emphasize them, adding annotations, improving the overall visual design. This also echoes the observations reported by Bigelow *et al.* in their study of "how designers design with data" [23], as well as the general observations made about design workflows in recent work about expressive visualization environments [24, 90, 97, 145, 89]. This kind of two-step workflow that

3. CREATING EXPRESSIVE NODE-LINK DIAGRAMS USING GRAPHIES

involves two independent tools suffers from multiple usability problems, as mentioned already.

The main problem is the artificial, and sometimes fuzzy, boundary imposed by the use of two standalone applications. When moving from the graph analysis tool to the vector-graphics editor, relations between visual marks and data are lost [24]. Any small modification that involves updating visual variables based on data attribute values entails going back to the first tool, regenerating the raw diagram, and then redoing all personalization edits in the second tool [23]. Another problem in the case of node-link diagrams is that, topological information having been lost in the vector graphics editor, editing operations in the personalization phase that rely on this information, such as, *e.g.*, adjusting the topology, or bundling and fanning edges, require tedious low-level geometry editing. In addition, a striking observation we made was how often users have to go back and forth between the visualization and the raw data in the graph analysis tool. This is consistent with Grammel *et al.*'s study [60], who observed that users switch back and forth between visual mappings and data attribute selection when constructing a visualization.

Relating our observations to Green's cognitive dimensions framework [61], we see that our users' design process is impeded mostly by significant *premature commitment* and by the *viscosity* associated with any small change, which is particularly pronounced when handling node-link diagrams. Based on these observations, we identified the following set of initial design requirements:

- R_1 : Avoid artificial boundaries between the data encoding stage and the visual design stage [97]: populating the canvas with nodes and links; specifying visual mappings; manually adjusting the layout and appearance of individual elements; adding static content such as annotations. All such actions should be seamlessly integrated into the same workspace, enabling designers to interleave them at will.
- R_2 : Enable designers to perform those operations using rapid, incremental, reversible actions whose effects on the objects of interest are visible immediately, following the principles of direct manipulation [159].
- R_3 : Enable designers to specify a wide range of on-node and on-edge encodings [119], for both categorical and numerical attributes.

In one of the later sessions with data analysts, our prototype had reached a sufficient level of maturity to conduct a preliminary evaluation. We asked three data analysts to perform a series of three specific tasks, first with their usual set of tools, then with Graphies. We worked in collaboration with a fourth analyst so as to make the tasks representative of their work. The other three data analysts were then asked to explore a dataset to answer a question that a client could have, and subsequently design a visualization to present their findings. The task was over when participants were satisfied with the node-link diagram they had made. We encouraged them to express out loud their actions when using Graphies. We also conducted a semi-structured interview at the end of the session to capture their general impressions. This preliminary study yielded encouraging results: all participants were satisfied with the produced visualizations, and never felt the need to resort to external tools. They particularly appreciated the ability to access data *in-situ*, directly from the canvas where they were

designing their visualization (R_1), and how rich (R_3) and easy-to-explore (R_2) the design space of visual encodings was.

At different stages during these sessions, data analysts expressed the need for additional features, which we took into consideration at various stages in the development of Graphies. For instance, in situations where they had to deal with networks similar in structure and content to other networks they had created visualizations of before, they wanted to be able to reuse those existing visualizations as starting points. We further discuss this feature in Section 3.3.4. They also made a set of related feature requests: be able to easily explore different designs and compare them; be able to animate between the stages of a design to better understand the differences between them; make it possible to export series of diagrams for the same network, arranged into an image gallery for storytelling purposes [89]. We derived the following two additional requirements from these feature requests:

- R_4 : Going beyond support for basic undo (R_2), enable multiple visualizations of the same network to co-exist, and let users easily fork and switch between them to explore alternative designs.
- R_5 : Enable smooth, animated transitions between these different visualizations, both as a means to help users keep track of changes and to produce animated node-link diagrams [11] that can support a rudimentary form of network storytelling [26].

3.3 Graphies

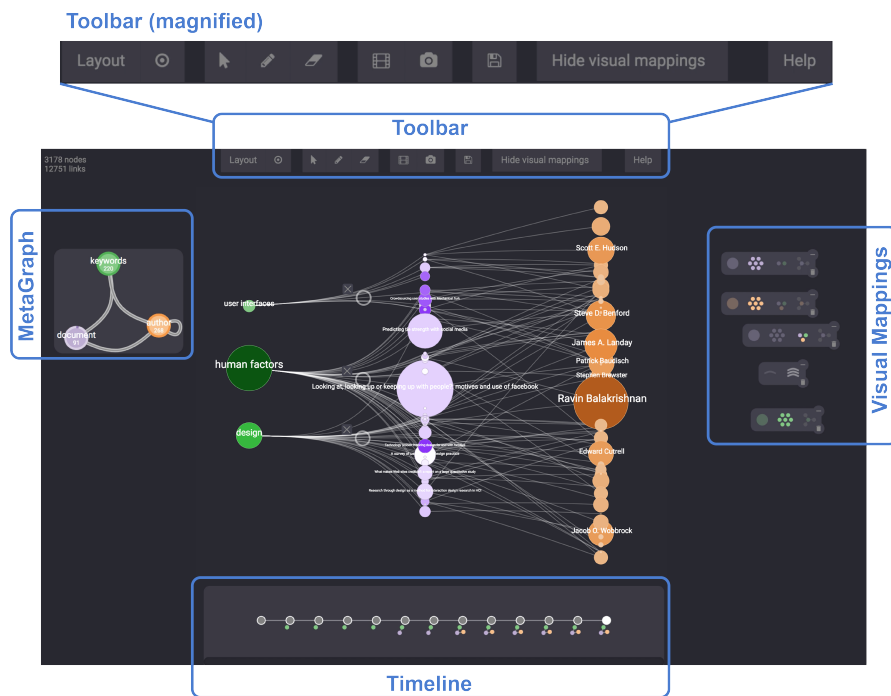


Figure 3.2: Graphies: main user interface components. The MetaGraph, Visual Mappings and Timeline are organized on independent layers.

3. CREATING EXPRESSIVE NODE-LINK DIAGRAMS USING GRAPHIES

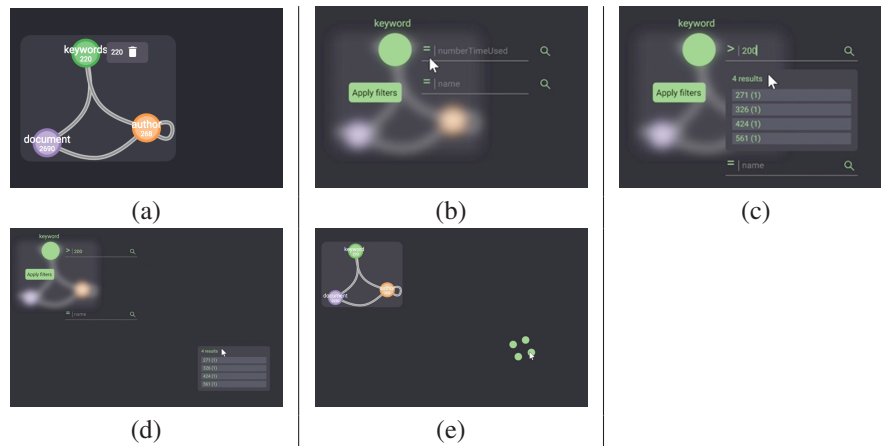


Figure 3.3: Applying a filter to nodes of type *keyword*: (a) there are initially 220 such nodes; (b-c) only selecting nodes whose attribute *numberTimeUsed* > 200 yields 4 results; (d-e) dragging-and-dropping those 4 nodes on the canvas.

As an expressive design tool, Graphies is primarily aimed at supporting the authoring of node-link diagrams made for communication purposes. In such contexts, users often know what elements or types of elements they want to show. They also have some idea about how they want to show them, but need to be able to quickly test different options. The core challenge is not to display all nodes and links, but rather to let designers build personalized views on subsets of the data using a flexible, iterative workflow.

Figure 3.2 shows Graphies' interface. Its main component is a zoomable canvas that holds the diagram. The MetaGraph on the left provides a summary of the full dataset (showing the different types of nodes and how they are connected), allowing users to incrementally populate the main canvas by dragging-and-dropping meta-nodes and meta-links directly in it. All interactions to modify the nodes' and links' visual appearance and spatial layout are performed by direct manipulations in the canvas or by invoking contextual widgets (shown in Figures 3.4 & 3.6). The Toolbar features icons that trigger global actions: optimize the layout, toggle annotation mode, save the workspace, export the diagram as a picture or video. Finally, a Timeline enables users to revert back to any past state of the diagram.

Implemented as a Web application, Graphies runs on a variety of devices: workstations equipped with mouse and keyboard, multi-touch devices, devices with support for pen + touch. On tactile surfaces, all navigation actions and manipulations of graphical elements can be triggered with touch gestures. When the device supports pen input, all interactions (including text entry) can be performed directly on screen (Figure 4.1). Such pen + touch environments are particularly well suited to designing node-link diagrams, as they make precise, arbitrarily-shaped selections of graph elements easy to perform. They also provide good support for free-form annotations, which can be useful when designing for communication purposes.

3.3.1 Populating the Canvas with Data

Users can import graphs from JSON files, or load previously-saved diagrams. The MetaGraph gets automatically populated with the different types of nodes and links

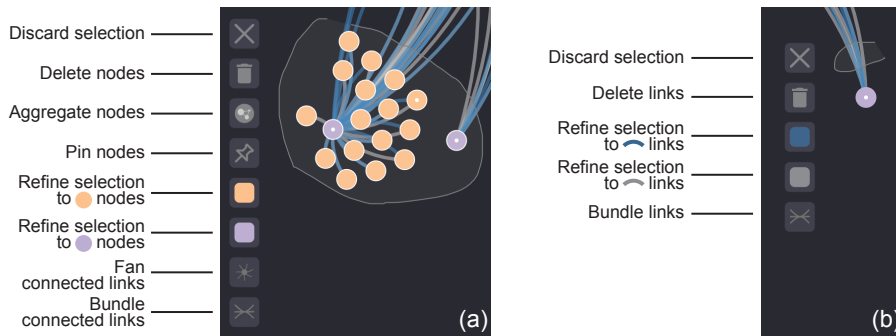


Figure 3.4: Contextual toolbars for node and link selections.

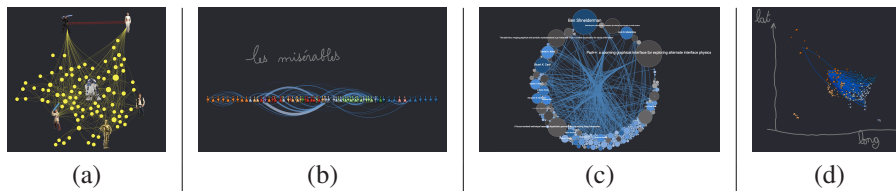


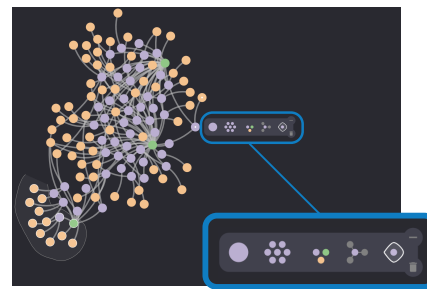
Figure 3.5: Sample node-link diagrams designed with Graphies: (a) Star Wars character co-occurrences; (b) character co-occurrences in Victor Hugo’s *Les Misérables*; (c) subset of SIGCHI publications and authors with keyword “*Information Visualization*”; and (d) flights in the USA. Annotations are used in (b) to add a title, and in (d) to show the axes used for node layout.

that are declared in the graph: for instance, in Figure 3.2, which shows HCI publication data, nodes are of type: document, keyword and author. Each node (resp. link) in the MetaGraph is thus an aggregated representation of all nodes (resp. links) of the corresponding type in the dataset.

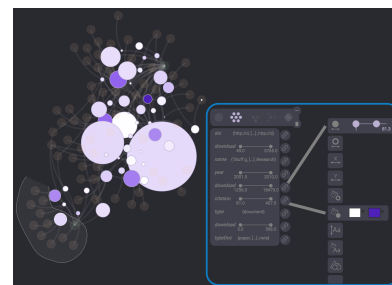
The MetaGraph allows users to populate the canvas without requiring them to interact with the raw data. It takes inspiration from OntoVis [156], while adding support for multivariate edges, and filtering. Similar to DataToon [89], the basic interaction consists of dragging a node or link type from the MetaGraph to the canvas to *populate* the latter with the corresponding elements. But unlike DataToon, which only enables *a posteriori* filtering of elements by direct selection after they have been dropped on the canvas, Graphies enables *a priori* declarative restrictions based on attribute values, before dragging elements. For instance, clicking on meta-node keyword in Figure 3.3 enables users to restrict the selection to those with the highest frequency of use (*e.g.*, those used at least 200 times). Once a filter is defined, users can drag either the whole result list or individual items to the canvas. Such *a priori* declarative filtering enables designers to deal with larger, more complex graphs featuring multiple attributes, both categorical and quantitative.

Dragging and dropping meta-nodes in the canvas only populates it with the corresponding nodes. To add links, users then have to drag-and-drop meta-links. They can be dropped on a node selection, in which case the canvas will be populated with the subset of links that have their source or target in that selection. Dropping on one specific node has a similar effect, adding links to this node only. This lets users

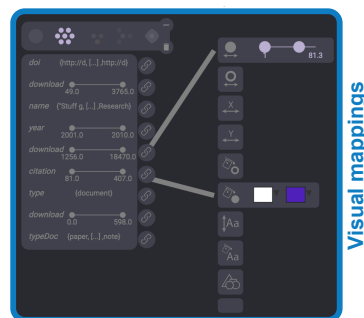
3. CREATING EXPRESSIVE NODE-LINK DIAGRAMS USING GRAPHIES



(a)



(b)



(c)

Figure 3.6: Visual mapping widget. (a) A long press on a node pops up a widget to set the scope of the mappings to be created. (b) Creating mappings by connecting attributes to visual variables, and adjusting ranges, all with immediate visual feedback.

start from a node of interest and make the graph expand incrementally, in the spirit of the “*Search, Show Context, Expand on Demand*” navigation paradigm [177]. Filters can be set on meta-links in the same manner as on meta-nodes.

This interaction model brings flexibility, letting users adopt either a bottom-up approach to the construction of their diagram by populating the canvas only with elements of interest after filtering; or a more classic top-down approach by populating the canvas with all elements first, and then defining filters for elements to be removed, as in [89].

3.3.2 Customizing the Layout






Node and link placement is a central concern when designing node-link diagrams. Graphies relies on D3’s implementation of the Barnes–Hut force-calculation algorithm [14] to optimize the graph layout. Force-directed layout algorithms have the good property of supporting incremental modifications to the graph. Users can drag and drop new nodes and links into the canvas, or remove existing ones, and the layout will smoothly adapt to these topological changes. The layout can be frozen by stopping an ongoing simulation, and the spatial density can be adjusted using a slider that controls the node repulsion force in the simulation. At any moment, users can also ask Graphies to optimize the layout again, with the option of forcing nodes into a radial layout (see, *e.g.*, Figure 3.5-c). Finally, they can adjust the position of any node freely by direct manipulation on the canvas. They can pin nodes as well, thus preventing the force simulation from moving them in subsequent runs.

Pinning is one of several commands exposed in the contextual toolbar that pops up when clicking on a node or making a lasso selection. The list of commands depends on whether the underlying selection consists of nodes (Figure 3.4-a) or links only (Figure 3.4-b). Beyond aggregate, pin and delete commands, the toolbar also makes it possible to further refine the selection. Indeed, some diagrams can make it extremely tedious to select only nodes or links of a given type in a particular region depending on the layout. With selection refinement, users can first delineate the region of interest on the canvas, regardless of the type of elements that lie inside, and then restrict the selection within those bounds to one type only.

Link-centric commands include interactive bundling [133] using a handle to manipulate the attenuation circle that parameterizes the bundle’s attraction force and location. In Figure 3.2, outgoing links from the three most-frequently-used keyword nodes have been bundled. In the same spirit, Graphies also makes it possible to interactively fan links [133] connected to a node, *i.e.*, to radially distribute them in a uniform way to decrease clutter.

3.3.3 Creating Visual Mappings

Multivariate graphs associate data attributes with nodes and links. Visualizations of multivariate graphs will typically show some of these attributes by mapping them to visual variables such as, *e.g.*, color, size, shape, stroke width, *etc.*

Figure 3.6 illustrates how such mappings can be defined with Graphies. A long-press on a node or on a link pops up the visual mapping widget. The first step consists of specifying the scope of the mappings that will be defined next. For nodes, the scope can be set (Figure 3.6-a) to: the node on which the widget was invoked ; all nodes of the same type ; all nodes regardless of their type ; the selected node’s neighborhood ; all nodes in active lasso selections on the canvas . Setting the scope then reveals the full visual mapping widget, as shown in Figure 3.6-b.

Data attributes are listed on the left, and visual variables on the right, along with domain and range information, respectively. Users declare visual mappings by connecting a data attribute with a visual variable. Both the domain and range of each mapping can be adjusted using interactive sliders and color selectors that behave according to direct manipulation principles. Changes made to visual mappings are directly propagated to the elements in the canvas, providing immediate visual feedback to users. Visual

3. CREATING EXPRESSIVE NODE-LINK DIAGRAMS USING GRAPHIES

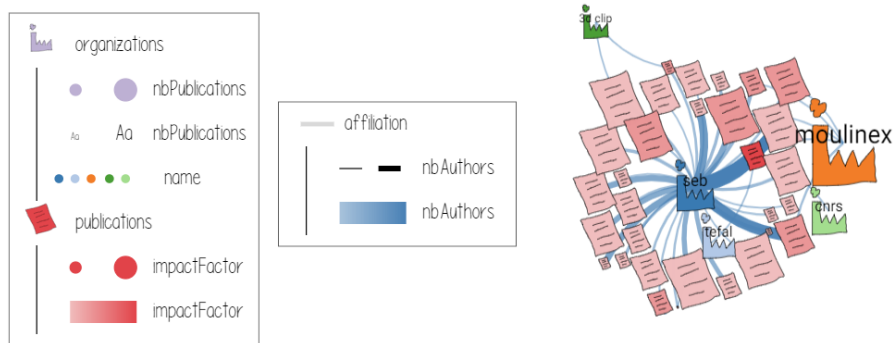














Figure 3.7: Legend generated automatically by Graphies for nodes and links.

variables that are not linked to a data attribute have the same value for all elements that fall in the scope of the mapping. This value can be adjusted using the same type of direct manipulation.

Graphies differentiates between categorical and quantitative data attributes. When the attribute is quantitative, the mapping between domain and range uses linear interpolation (in HSL space for color). When it is categorical, attribute values are mapped to discrete values evenly distributed in the visual variable's range (choosing from predefined schemes in the case of color hue).

Graphies supports a rich set of visual variables, giving users much flexibility in how they can represent the data. Users can map node attributes to the following visual variables: spatial position , size , stroke width , stroke color , fill color , shape , label size  and color . Mapping attributes to spatial position makes it straightforward to create attribute-driven layouts [161, 184]. For instance, nodes in Figure 3.2 are arranged in columns by mapping type (a categorical attribute) to the node's x-coordinate. Label size and color are managed as visual node properties. By default, labels are shown only when hovering. They can also be made always visible by setting the label size to a constant value. Finally, the node shape can be chosen among a set of basic geometries: circle, square, triangle. It can also be sketched, in the spirit of [190, 89] (Figures 3.5-b & 3.7), or replaced by a bitmap (Figure 3.5-a).

Similarly, users can map data attributes to link variables: stroke width , stroke color , curvature  (which has recently been proposed as a means to encode data on links [133]), and texture . The latter can be based on a sketch or a bitmap. In each case, the texture consists of the input rendered in a repeating pattern along the link.

The variety of possible visual mappings on both nodes and links means that designers might want to create a legend to accompany their diagram. This is particularly relevant in the case of expressive designs intended for communication purposes. As creating such legends can be a tedious task, Graphies features a legend generator (see Figure 3.7) that automatically builds a summary of visual mappings that can later be manually edited. Mappings that have a global scope come first, followed by mappings that apply to a subset of elements only. The legend can also serve as a quick selection tool: clicking an item will select all nodes or links the corresponding visual mapping applies to.

3.3.4 Supporting an Iterative Design Process

A key characteristic of expressive visualization design environments is not only to provide designers with a wide range of options to visually encode data, but to enable a flexible design workflow as well. The environment should let designers quickly try multiple alternatives and iterate on their diagram, in an *exploratory design process*.

As discussed earlier, this is primarily supported by enabling designers to perform all steps of the workflow in a single environment without imposing artificial divisions between, *e.g.*, the data encoding stage and the visual design stage [97]. But additional features contribute to facilitating an iterative design process in Graphies, as detailed next.

One important design choice we made in Graphies was about the visual mapping evaluation strategy. Visual mappings can be seen as rules that apply a set of styling instructions to a selection of elements as in, *e.g.*, the Cascading Style Sheet language (CSS). The two main options to handle such sets of rules in Graphies were: a) to always enforce mappings; or b) to apply them on-demand only. While option (a) will typically yield a higher level of consistency, it does this by restricting the designers' options, or resolving conflicts, or both. Option (b), on the other hand, applies mappings blindly, without imposing any restriction. However, it will also override values previously set by other visual mappings that apply to (some of) the same graph elements. As flexibility is key to expressive design, we eventually opted for option (b). Active mappings are not enforced continuously by the system, but rather applied to the associated selection of nodes or links on-demand, implicitly, when designers interact with the corresponding widget. As it is important to be able to adjust the mappings, they can be accessed quickly from an independent layer drawn on top of the canvas. All mappings reside in this layer indefinitely (unless explicitly deleted), even if they conflict with other mappings. That layer can be hidden, or the mappings can be minimized individually to limit clutter.

Beyond the adjustment of existing mappings by direct manipulation, the exploratory design process is also supported by the possibility to backtrack to previous states. In a similar setting, Lyra's authors observed that the absence of undo had a negative impact on users' willingness to explore alternative designs [145]. Graphies not only supports basic undo, but branching as well, enabling designers to explore several alternative designs in parallel. A new state is created when: nodes or links are added/removed from the canvas; and when visual mappings are modified. Past states are accessed from the timeline (Figure 3.2). Branches appear as multiple connected tracks in the timeline, as can be seen in Figure 4.1. Each small circle corresponds to a past state. Hovering a circle pops-up a preview thumbnail image rendering of the diagram in that state, to make it easier for users to find the appropriate past state. Resuming editing from a past state will automatically create a new branch, and the corresponding track in the timeline. Loading a saved project restores the entire history, including branches.

3.3.5 Exporting Visual Designs

Diagrams designed with Graphies can be exported as SVG files, but more elaborate export functions relevant to the use of node-link diagrams for communication purposes have also been implemented. Designers can export selected keyframes (stages) as an image gallery, with the possibility to add textual annotations to each image in order to tell a story [10, 89].

3. CREATING EXPRESSIVE NODE-LINK DIAGRAMS USING GRAPHIES

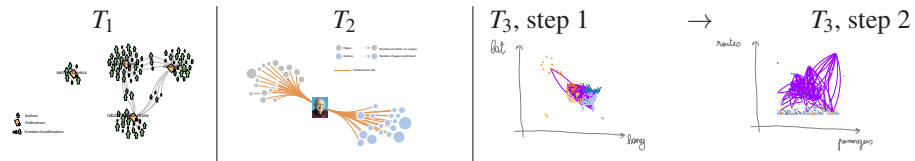


Figure 3.8: The three target visualizations that participants had to reproduce during the experiment.

The evolution of a graph over time [11] can also be animated and exported as a video clip (MPEG-4). Here again, designers select stages from the timeline, that they use to populate the track of a simple video editor. These states, which can be reordered, define the keyframes of the animation. Graphies then renders the video clip by smoothly interpolating between those keyframes.

As discussed in Section 3.2, such animations can be useful when telling stories about the data: incrementally bringing data to make a point step-by-step, showing complementary perspectives on the data, showing the actual evolution of a graph over time, or even documenting the incremental construction of the diagram. To further support the use of these diagrams for communication purposes, Graphies enables designers to add freeform annotations to the canvas, as illustrated in Figures 3.5-b & 3.5-d.

3.3.6 Implementation

Graphies is a Web-based application, implemented with D3 [25] and `paper.js`. The former handles SVG UI widgets, as well as many computations (visual mappings, force-directed layout, *etc.*), while the latter is used to draw the node-link diagram. We draw the diagram using 2D primitives directly in an HTML `<canvas>`, as opposed to manipulating an SVG DOM hierarchy, for the sake of performance. The two UI components are seamlessly integrated using CSS rules and input event redirections. Edit history management and MPEG-4 video export are handled on the server side.

3.4 User study

The interaction model proposed in Graphies differing significantly from that of other graph visualization tools, we conducted a *first-use study* (using a term coined by Hartmann *et al.* [68]) to gather empirical observations about: 1) users' ability to reproduce existing expressive visualizations using Graphies; and 2) how effective the tool is at supporting the creation of expressive designs.

3.4.1 Procedure

Participants start the experiment by following a tutorial ($\leq 30mn$) introducing Graphies using a dataset about interactions between Star Wars characters (that dataset was not used afterwards). The operator follows a script, demonstrating a series of features in the same manner to all participants. Each time a new feature is introduced, he asks participants to use it on a different example to make sure they have understood how to use it. The list of features is: *add all nodes of a given type; add only nodes that match an attribute value filter; add all links of a given type to a specific node; then to a selection of nodes; change the visual variables of a set of elements; create a visual*

mapping; use the selection menu to remove some nodes and links; bundle or fan a selection of links; add keyframes to the video editor and play the resulting video; pan, zoom and reset the view; make some annotations.

Participants then have to reproduce each of the three visualizations in Figure 3.8, which use visual presentation techniques from the recent literature. These are *re-creation* or *reproduction* tasks, conceptually similar to those used to evaluate many recent visualization authoring tools, *e.g.*, [145, 90, 97, 190, 89, 132]. Participants are presented with a printed image of each visualization, accompanied by a short text listing the main steps (what attribute to show, what visual encoding to use, *etc.*), in order to avoid ambiguities. The three tasks are always presented in the same order.

The diagram in task T_1 shows documents about four product categories, as well as their authors. The two node types use different sketch-based glyphs. The size of author nodes indicates how many documents they have authored.

Task T_2 is about reproducing a co-authorship network. The network consists of authors and publications. Fill color encodes node type. Size encodes the number of articles published (author nodes), or the number of co-authors (publication nodes). Emphasis is put on central actor Ben Shneiderman by 1) using his picture to decorate his node (as in, *e.g.*, Vizster [69]); and 2) forcing all publication nodes on the left and all co-author nodes on the right, bundling edges interactively [133] on each side.

Task T_3 is inspired by PivotGraph [184]. Participants have to create an animation between two views on the same US air traffic network. In step 1, airport nodes are laid out according to their geolocation attributes (latitude/longitude). Links show routes between south-east and north-west airports only. Stroke-width encodes the number of carriers on a route. In step 2, the same airport nodes are laid out in a scatterplot-like manner: the number of transiting passengers is mapped to the x-axis, the number of routes connecting this airport is mapped to the y-axis. In both steps, participants have to use free-form annotations to indicate the axes.

Finally, participants are asked to perform an open-ended task (T_{free}): design a visualization of their choosing about publications in the field of Information Visualization. The visualization should show the most prolific authors and their articles, emphasizing highly-visible articles. Instructions make it clear that *highly-visible* should be interpreted here as *having a large number of citations*.

3.4.2 Participants & Apparatus

Ten volunteers (1 female), aged 21 to 42 year-old (average 27, median 25), participated in the experiment. Three of them were undergraduate students (P2, P7, P9), three were software engineers (P4, P6, P8), three were PhD students in HCI (P0, P1, P3) and one was a PhD student in Information Visualization (P5). We conducted the experiment on a Microsoft Surface Book 2 13" (3000 × 2000 pixels), equipped with an Intel Core i7 processor, 16GB RAM, and an NVIDIA GTX 1050 (2GB) graphics card.

3.4.2.1 Data Collection

We used screen capture software and audio-video recordings. We also timed users for each task. At the end of each task, we captured the visualization they created. A single

experimenter was present for each session. When users were performing trials, the experimenter retrieved comments and feedback using a pen and a paper notebook.

3.4.3 Results

Figure 3.9 shows all visualizations produced by the participants, using the last frame of the animation (step 2) for T_3 . All study material is made available on the project's Web site, including datasets, instructions, and the animations created by participants in T_3 : https://hugoromat.github.io/graphies/user_study.html and in Appendix A.1. All participants managed to properly reproduce the target visualizations from Figure 3.8 *on their own*. Indeed, once the tutorial completed, the operator never gave any additional explanation about Graphies' features. He only answered the few questions participants had about how precise their reproductions had to be, telling them that the goal was to achieve a visualization that would *show the same data*, using the *same visual encodings*; but that they should not try to accurately replicate the spatial layout and link bundles, or use the exact same sketches (for nodes in T_1) and colors.

To give an indication, task completion times per task were (in seconds): $353 \pm 90s$ (T_1), $498 \pm 81s$ (T_2) and $582 \pm 59s$ (T_3). But more importantly, all reproductions, regardless of the task and participant, feature the correct number of nodes and links. Given the above instructions, there are obviously variations in the local placement of elements, but the global layout strategy matches that of the templates, as do colors and other visual variables (compare Figures 3.8 & 3.9).

Participants spent $463 \pm 120s$ on the open-ended task (T_{free}) where they had to *create a design of their own*. What is striking when looking at the rightmost column in Figure 3.9 is the diversity of visualizations that participants created from the same dataset. While 4 participants exclusively made use of the *size* visual variable in their mappings, all others used additional channels, including position, color, transparency, and node shape (sketched glyphs). As expected, several of them also performed manual layout adjustments.

3.5 Discussion, Feedback and Future Work

The diagrams produced in T_{free} (rightmost column in Figure 3.9) demonstrate that users can be very creative when designing node-link diagrams. This diversity is also an encouraging result for Graphies, as participants were able to create those visualizations in a very short amount of time. Overall, these observations suggest that the approach has good potential; even more so when considering that our participants, while they all had a background in Computer Science, were not professional visualization designers. Indeed, with the exception of P5, none of them had significant experience in Information Visualization. At best, the three PhD students had followed an introductory course.

Beyond these high-level observations, we noted some interesting behaviors and gathered feedback from study participants. We discuss these along with additional feedback from the data analysts who were involved in the design process (Section 3.2), that was gathered after the study. Some of this feedback has already been taken into account, while other features are left as future work, as detailed next.

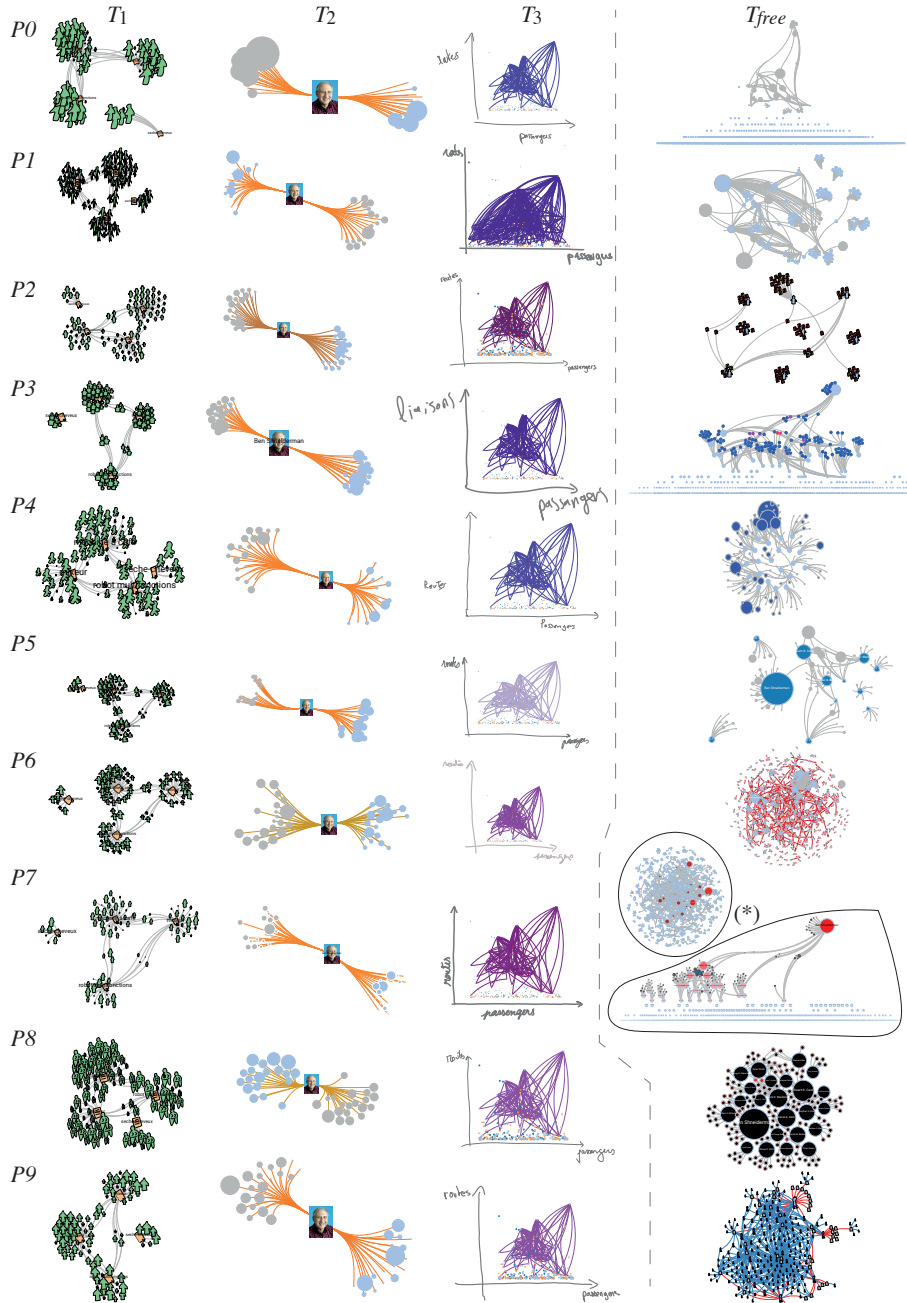


Figure 3.9: Visualizations created by all ten study participants (one line per participant, zoom-in to see details). From left to right: T_1 , T_2 , T_3 (step 2) and T_{free} . (*) P7 asked for the possibility to make two complementary visualizations for T_{free} .

3.5.1 Observations



All participants did not follow the same workflow, confirming the need for flexibility in the authoring process already observed in prior work [97, 23, 24]. Some participants started by populating the workspace with all relevant elements, spatially arranging them before creating any visual mapping; while other participants quickly moved to creating visual mappings, intermingling actions to populate the canvas, arrange elements, and change their visual appearance.

Our interaction model aims at making most actions accessible directly from the main canvas, and at enabling users to easily iterate on their designs. This entails keeping all visual mapping widgets close at hand, on a dedicated layer. While we have dedicated much effort to designing a compact widget, the fact that it gets replicated each time a new visual mapping scope is defined means that this layer in the workspace may get cluttered quickly. Users can move, minimize and even discard the widgets, but our observation of study participants suggests that users do not take the time to organize their workspace. As the study was relatively short, it would be interesting to see if this remains true for longer work sessions and more frequent use of the tool. If it turned out to be the case, mechanisms such as automatically minimizing mapping widgets and moving them to the edge of the workspace could be investigated. However, this should be done with caution, as users might then experience difficulty keeping track of what elements in the canvas a given mapping widget applies to (its scope).

3.5.2 Feedback from Study Participants

Several participants spontaneously commented positively about the richness of visual mappings. So did they about the possibility to filter elements *before* adding them to the canvas (see rationale in Section 3.3.1), especially participants who had started by adding all elements of a given type before backtracking, as their diagram had become too complicated.

Participants raised a few issues too. Related to the mapping-widget clutter problem, one participant said that he would have liked to be able to relate widgets to the elements in their scope. We had actually discussed this issue during the design phase. But while we can use transient highlighting to emphasize the scope of mappings one at a time (using, *e.g.*, legends - Section 3.3.3), we have not yet found a satisfying solution that would work for all widgets simultaneously without causing considerable clutter.

Still related to visual mapping widgets, some participants were a bit confused about the meaning of the trash icon  in the bottom-right corner of the widget (Figure 3.6-a). They expected it not just to discard the widget, but to actually cancel the effects of the mapping on the visual appearance of elements in its scope. Such a behavior could indeed make sense, but is not compatible with the visual mapping evaluation strategy we chose. The system applies mappings on-demand only, *i.e.*, implicitly, whenever their parameters are adjusted; it does not enforce them continuously. With this evaluation strategy (see Section 3.3.4 for the details and rationale), discarding the rules has no impact on the diagram. To alleviate this source of confusion, we switched to a cross icon  as a more appropriate symbol.

One participant asked if Graphies could suggest appropriate mappings [60]. Visualization grammars such as Vega-Lite [146] do this, and it would indeed be interesting to study such a possibility in future work.

3.5.3 Feedback from Data Analysts

The data analysts involved in the design process were invited to participate in one last session organized after the user study. They made several feature requests that also relate to bootstrapping the design workflow.

One straightforward request was to include a global search function to quickly access specific elements whose name is already known, when the designer is highly familiar with the dataset. More interestingly, in the opposite context of working with a dataset that is not necessarily very familiar to the designer, the analysts asked if, in addition to suggesting visual mappings, Graphies could also suggest a subset of nodes and links to start populating the diagram based on metrics such as centrality or similarity. They indicated that this would be, again, particularly useful in the context of unfamiliar datasets. Such features, which echo Crnovrsanin *et al.*'s recommendations for network navigation [39] and DataToon's suggestions based on the detection of structural patterns [89], can also help users who are intimidated by a blank canvas [145].

Visual mapping and subgraph suggestions relate more generally to the concept of templates, which is another feature we discussed with analysts. They often use datasets that share the same structure (node and link types, associated attributes) to produce different visualizations across projects. It makes sense to reuse mappings defined in earlier projects, at least as a basis for the new diagrams. In this context, the idea of reusable templates arose. In Graphies, templates correspond to a set of visual mappings that each have a fairly general scope, typically a specific type of node or link (Figure 3.10). They essentially play the role of reusable stylesheets, as skilled users can write with Graph Coiffure [165]. By calling a template on datasets that share the same general structure, designers can quickly populate their canvas with nodes and links whose appearance has already been predefined according to the rules in the mappings.

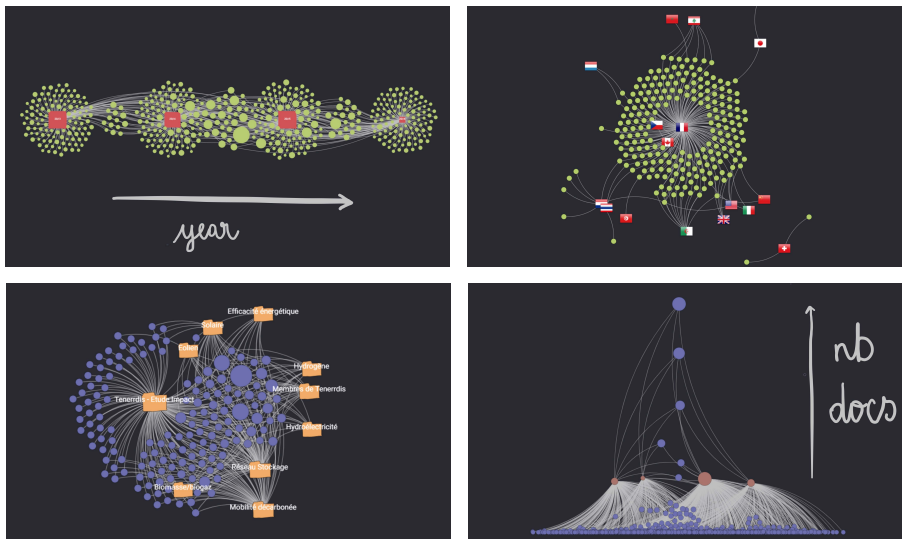


Figure 3.10: Different templates for a dataset featuring multiple node types, including: authors, organizations, documents, thematic.

Finally, as the direct manipulation of nodes and links is a central activity when designing expressive node-link diagrams, it would also be relevant to include subgraph selection

3. CREATING EXPRESSIVE NODE-LINK DIAGRAMS USING GRAPHIES

and manipulation techniques by McGuffin & Jurisica [107], as they would make tasks such as node selection and layout customization easier to perform.

Using motion as an encoding channel in node-link diagrams

The work presented here was published at CHI 2018 and INTERACT 2019 [135, 137]. During this project I implemented the WebGL version of Animated Edge Textures, and conducted the first study. Dylan Lebout implemented the SVG version, and conducted the second study. The paper was written collaboratively by all authors.

The amount of literature on graph visualization is considerable [20, 63, 179]. Beyond research on layout algorithms for node-link representations, the community has explored several aspects pertaining to the visual encoding of data attributes on nodes [184] and links [80, 72]. Focusing on links only, data attributes are usually encoded using the following visual variables: color, opacity, stroke thickness and stroke pattern. In the case of dynamic visualizations, this relatively limited set of variables can be augmented using animations [35].

Animating directed edges in node-link diagrams is typically achieved by having sequences of small glyphs, which we will call *particles*, dynamically move along the links [9]. The visual effect is that of an *animated texture* applied to links. In some cases, there is a one-to-one correspondence between particles and entities actually traversing the network, as for instance in the case of traffic visualizations where each particle represents one vehicle [33, 150]. In other cases, there is no such correspondence: the particles are elements of a cyclic animated texture, whose visual properties (pattern, speed, frequency – see Figure 4.1) encode more abstract data attributes. Examples of the latter include visualizations of telecommunication networks (attributes such as, *e.g.*, failure rate, lag, bandwidth) or import/export of goods between countries [38].

This chapter seeks to provide a deeper understanding of the perceptual quality of animated edge textures, as well as to explore more systematically their design space. After providing a motivation for our work, we introduce a model and associated Web-based framework for generating animated edge textures, and illustrate its capabilities using different examples of visual mappings. We then report on an initial evaluation of particle properties in terms of visual perception. The results suggest that the three motion variables defining animated edge textures - speed, frequency and pattern

(Figure 4.1) can be used individually for encoding data attributes. We conclude with a discussion of the limitations of this first study, and an outline of the next steps to pursue this line of research.

4.1 Background and Motivation

The number of visual variables available for edge attribute encoding in node-link diagrams is limited, and while there are opportunities for combining those variables, they are not completely orthogonal. For instance, color and opacity will, to some extent, mutually interfere. Stroke width, color and opacity will also all influence the saliency of an edge. The practical range of values for these variables is also fairly limited. For instance, wide strokes are likely to cause visual occlusion of elements in the background, or of other edges in dense graphs. Beyond the question of efficiency of the encoding in terms of visual perception, issues related to the aesthetics of the visualization can quickly arise, that go beyond pure considerations of graph layout [183].

Taking the example of a statistical dataset about domestic flights in the USA,¹ the number of featured edge attributes far exceeds the number of visual variables. As illustrated in Figure 4.2, the number of airlines operating on a route could be encoded using the stroke width; the daily number of flights could be encoded using a stroke pattern made of dashes whose spacing is a function of that value; and the daily number of passengers on a given route could be encoded using color brilliance. But encoding more attributes would be challenging.

By gaining a better understanding of what other visual variables can be used to encode data in node-link diagrams, our goal is *not* to find *better* visual variables, but rather to identify *alternative* encodings, widening the design space of visual mappings in node-link diagrams, in the same spirit as Henry Riche *et al.* [133], who explored the design space of link curvature to represent attributes such as edge directionality. In this

¹Obtained from <https://www.transtats.bts.gov>.

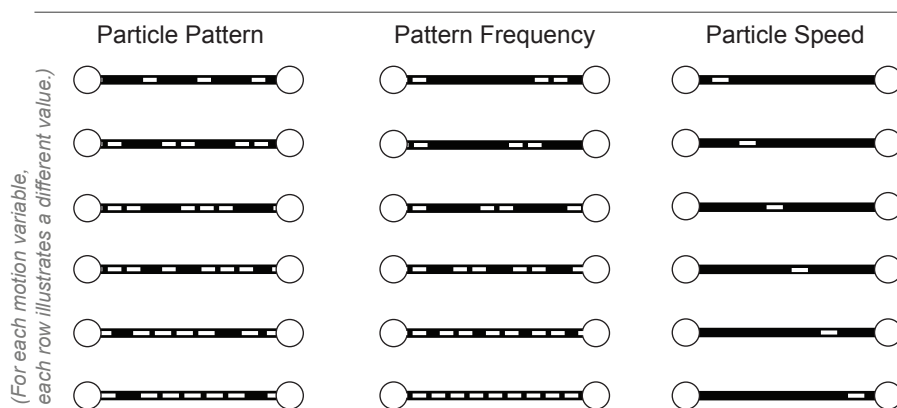


Figure 4.1: The three variables defining the dynamic behavior of particles in animated edge textures: the *particle pattern*, or rhythm, is the sequence of particles that gets repeated cyclically; the *pattern frequency* corresponds to the firing frequency of the particle sequence; the *particle speed* corresponds to the velocity of particles on screen.

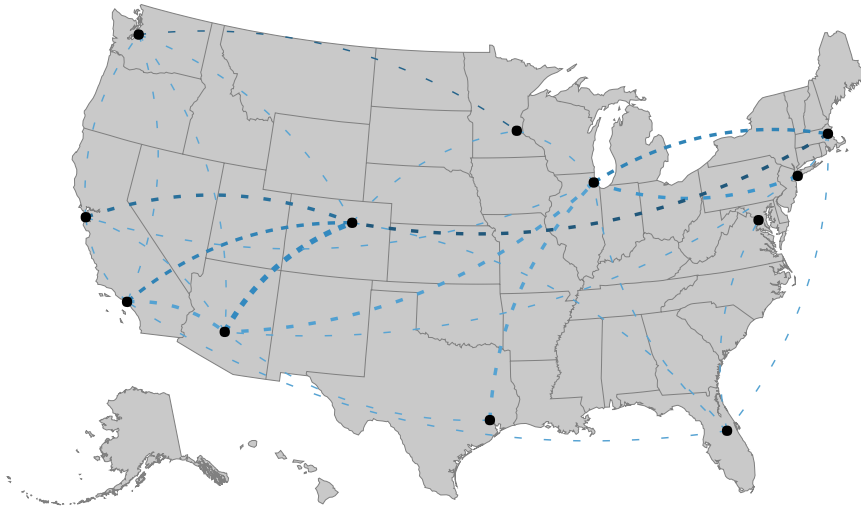


Figure 4.2: Mapping three data attributes on three static visual variables.

work we focus on motion, which has been identified by Bartram, Ware and colleagues as “*hold[-ing] promise [...] as a perceptually efficient display dimension*” [16].

Coming back to the above example about air traffic statistics, the introduction of motion as a means to encode data widens the space of possibilities in two ways. It can either be seen as a way to encode additional data attributes: for instance, the average speed on a route can be encoded using particle speed. Or it can be seen as a way to encode some data attributes using a different visual variable deemed better suited: for instance, mapping the daily number of passengers to particle speed can give a sense of route “throughput” that color brilliance might not convey as effectively. Motion-based encodings can also provide alternatives when visual variables such as color are not available, for example when the underlying layers in the visualization already make use of color to encode other data, or on electronic ink (monochrome) displays.

Motion has potential as a means to encode edge data attributes in graphs, and more specifically in *directed* graphs, as motion along an edge explicitly suggests a specific orientation of that edge. But possible usages of motion in network visualization need to be better understood. Our goal is to define a design space of motion-based data encoding in node-link diagrams, and to enable the in-depth empirical assessment of their characteristics in terms of perception. To this end, we introduce a framework called Animated Edge Textures. Before introducing this framework, we discuss findings from the literature that are relevant to our research.

4.1.1 Motion in Human-Computer Interaction

While animations, including motion, should be used with caution [175], there are situations where they have proven useful. Studies about the potential of motion in HCI date back to the early 1990s with seminal work such as Ware *et al.*’s investigation of motion as a means to attract user attention [182]. Motion is also useful in scientific visualization [81], as the processes visualized often involve the dimension of time [53]. Very early, Bartram started investigating motion as an “*abstractly codable dimension*

in its own right" [15], a direction explored in other projects [16, 17, 98, 99], and in which we situate our own work.

In information visualization, animations might not be effective as an exploration aid, but can be powerful tools for presentation [53]. Motion can be useful for filtering and brushing [16, 50, 181]. It can help emphasize spatial relationships, explain functionality, illustrate causality [113, 17]. It is also used extensively in flow visualizations [66, 81, 109] and representations of other scientific data such as cosmological particles [65].

Motion has been observed to evoke affect [52] and has an impact on the perceived aesthetic qualities of a visualization. This likely plays a role in the use of particle-based animations in some design-oriented visualizations [38, 41, 78, 117, 126].

4.1.2 Motion Perception

The body of literature on motion perception in experimental psychology is very large [153]. While it is not our intent to report them exhaustively in this section, we highlight key findings on motion perception that makes this encoding compelling to explore for information visualization researchers. We also motivate the need for perception studies closer to visualization applications.

Early research demonstrated the ability of simple motions to communicate complex or nuanced behaviors [70]. A crucial insight for information visualization is that motion belongs to the limited set of visual encodings that is perceived preattentively [174], *i.e.*, detected before focused attention. Motion can make moving shapes naturally "pop out" in visualizations. Orban *et al.* [122] observed that different motion velocities are easier to discriminate in the central visual field, the just-noticeable difference in velocity increasing as eccentricity increases, particularly so for low reference velocities.

The perception of motion in conjunction with other encodings such as color has been studied in prior work showing, for example, that a conjunctive search for motion and color is serial rather than parallel [118], whereas a conjunctive search for motion and form is parallel [108]. Questions related to conjunction in guided search have been further studied by Driver and colleagues. They observed that it can be parallel under some circumstances, but that out-of-phase motion across elements had a significant negative impact on search performance [44]. They also observed that the search for a target with a different orientation was faster among moving distractors than stationary ones, provided the target's orientation is not too close to that of non targets [43].

Beyond the Gestalt law of common fate, which states that elements tend to be perceived as a group if they move together, another interesting observation is that moving objects can capture attention [54]. Scimeca and Franconeri [152] highlight the main findings related to tracking multiple objects (or groups of objects) in motion. Multiple studies have investigated our ability to detect changes in direction [170], speed [105, 83] or both [62, 79]. The role that the color and luminance of elements play in the perception of motion direction and velocity has also been studied extensively, as summarized by Weiskopf [185].

The above psychophysiological-level findings can guide and inform the research and design of novel visualization techniques, but they cannot, alone, answer higher-level questions related to the perceptual efficiency of motion-based visual encodings in node-link diagrams. Indeed, the use of motion on graph edges is quite specific. The

considered elements (particles) only move along predefined visual paths. Particles that belong to the same edge get perceived as a group not only because they have a common fate but also because they visually materialize a first-class entity: edges. The nature of the movement and the number of different directions, considering that edges have different orientations and that they can be curved, is very particular compared to the conditions evaluated in the above-cited studies. Empirical studies that focus on these conditions are necessary to get a clear understanding, and to quantify the limits, of motion-based data encodings in tree and network visualization.

4.2 Animated Edge Textures

Our model of animated edge textures consists of three main motion-related variables describing the behavior of particles flowing along links, as illustrated in Figure 4.1: the *pattern* of particles, which can be seen as the rhythm at which they get fired; the *frequency* of this pattern, and finally, the *speed* of particles along the link. Each variable can be mapped to different data attributes. They can also be combined with other *visual variables* that define the appearance of both the particles (*e.g.*, their color, their shape) and of the link itself to encode additional edge data. In this section, we describe the design space of animated edge textures, and introduce an API for mapping data attributes to visual variables in this design space, along with a prototype implementation in WebGL.

4.2.1 Model

The simplest way to animate particles along an edge would be to define a repeating stroke pattern, as one can do in vector graphics editors such as Adobe Illustrator, and have this repetition of the pattern animate by gradually increasing its start offset. However, this approach is limited in terms of expressive power. It also fails to expose all motion variables for direct mapping with data attributes. We seek to strike a balance between simplicity of the model, expressive power, and ease of mapping between data attributes and visual variables.

Our model is based on the following core concepts:

- The *link* itself, which encodes the path geometry of the corresponding edge (for a given layout). The link can be seen as a tunnel through which the particles flow. The visual appearance of that tunnel is defined by the static visual variables usually associated with links in a node-link diagram: color, opacity, stroke width.²
- The *particle emitter* fires particles from the source node of an edge according to the particle pattern and pattern frequency. A particle emitter defines the properties of particles it fires: their shape, color, size, and speed.
- The *particles* themselves, which are the individual glyphs that flow along the link.

Two additional concepts are introduced to enable more elaborate motion-based encodings:

²A stroke pattern other than solid should be avoided, as it would visually interfere with the particles flowing along the link.

- *Gates*, placed at arbitrary positions along an edge, can change one or more variables of the particles flowing along that edge, including their color, opacity, size and speed. Indeed, once emitted, particles are independent entities whose properties can be altered as they progress along the link. Gates perform those alterations smoothly: the variable gets interpolated between the old and the new value over a short span centered on the gate, so as to avoid abrupt changes that would be aesthetically unpleasant and visually disturbing. Several examples in the next section use gates to alter particle variables. Variations in particle speed encode specific data attributes in Figure 4.3. Variations in particle opacity minimize legibility issues in Figure 4.4. Variations in particle color can be used, *e.g.*, for aesthetic purposes, or to visually illustrate the notion of transformation of what flows through the edges.
- *Tracks* enable particles to flow along multiple parallel paths inside the same link tunnel. All tracks that belong to the same link share the same emitter, for the sake of model simplicity. Nevertheless, the model allows for multiple links between two nodes, thus enabling different particle patterns on parallel tracks, as is done in Figure 4.5.

4.2.2 API

We have designed a Javascript API to map data attributes to any edge variable, including all three motion variables (pattern, frequency and speed). The following code fragment illustrates the simplicity of our API. It shows how a graph is loaded and laid out (lines 1-5), similar to D3 [25] or cola.js [47]. Lines 6-7 set the parameters controlling the particles' visual properties, while lines 8-10 set motion properties.

```
1 d3.json("data.json", function(json) {
2   var force = flownet.force("#aFrame", 800, 600, "#FFF", 0)
3   .nodes(json.nodes)
4   .links(json.links)
5   .create_layout()
6   .particles("color", "#CCC")
7   .particles("size", 2)
8   .particles("pattern", [0.0, 0.5, 0.75])
9   .particles("frequency", 0.4)
10  .particles("speed", 3);
11 });
```

Mapping these parameters to actual edge data attributes is achieved using anonymous functions, as shown below (d being a link instance, following the same convention as in D3). In this example, particle speed is set to be proportional ($\times 10$) to the value of edge attribute val on each link:

```
.particles("speed", function(d){return d.val * 10;})
```

Temporal patterns are specified as arrays of floats in $[0, 1[$. Each item in the array corresponds to a particle in the sequence defining the motif. Line 8 in the above code

fragment defines a pattern made of three particles, fired by the emitter at the beginning (0.0), middle (0.5) and three quarters (.75) of each cycle. The actual timing depends on the pattern frequency for each edge. In this case, the pattern frequency being set to 0.4 Hz, the three particles are fired at 0s, 1.25s and 1.875s, repeating every 2.5s. Patterns can be assigned to individual edges based on any attribute, as shown below with a test of attribute `cat`:

```
.particles("pattern", function(d){
  if (d.cat == "dualLink"){return [0, .33, .66];}
  else {return [0, .2];}
})
```

In the example below, either one or two tracks are created, depending on the value of attribute `cat` for each edge:

```
force.tracks("count", function(d){return (d.cat == "dualLink") ? 2 : 1;});
```

Alterations to visual and motion attributes when passing through gates are specified by passing one additional parameter to function `particles()`. In the example below, particles are colored green on the first half of the link, and red on the second. Similarly, the speed of particles is 10 times the value of edge attribute `val` during the first 80% of their course, and increases to 20 times that value on the remaining 20%:

```
.particles("color", "#0F0", 0)
.particles("color", "#F00", .5)
.particles("speed", function(d){return d.val*10;}, 0)
.particles("speed", function(d){return d.val*20;}, .8);
```

4.2.3 Prototype Implementation

We have developed a prototype Javascript implementation of our model, called FlowNet. FlowNet can be used in conjunction with D3 to load and prepare the data for visualization, and graph layout libraries such as `cola.js` to compute node placements. Rendering takes place in an HTML5 `canvas` element using `Three.js` [42]. Using SVG for rendering would have enabled a tighter integration of FlowNet with D3, but our early tests showed that this solution would not have been able to handle large numbers of particles. Using WebGL enables us to take advantage of GPU rendering, vertex and fragment shaders being able to manage larger amounts of particles.

4.3 Examples of Use

We illustrate different combinations of motion variables using subsets of the air traffic statistics dataset introduced earlier. The data are not about individual flights, but are rather aggregated statistics over all airlines on each route, thus forming a network that features a rich set of attributes for both nodes (airports) and links (flight routes). Animated versions of the examples shown here are available in the companion video

and Website.³ For the sake of clarity, we selected very small data subsets, which lend themselves better to static representations for inclusion in this thesis. As stated earlier, motion variables should not be seen as systematic replacements for other visual variables such as color and stroke width, but rather as alternative visual variables that widen the space of possibilities and have their own strengths and weaknesses.

4.3.1 Encoding a single attribute

The simplest example consists of encoding a single edge attribute using a motion variable. Both numerical and categorical data can be encoded using motion. In the air traffic dataset, numerical attributes associated with each edge include the number of flights on the corresponding route, or the average departure delay on that route. Categorical attributes include the type of route (passenger transportation, mail, or freight).

4.3.1.1 Encoding a numerical attribute.

Numerical attributes can be directly mapped to speed or frequency. One guiding principle in the selection of which motion variable to choose in the design space is to pay attention to the semantics inherently conveyed by that variable. For instance, when seeking to encode the number of flights on a route, frequency is a more appropriate choice than speed, as users will tend to interpret a higher frequency, which entails a higher particle density on the link, as a larger number of flights. Conversely, speed may convey the notion of a more efficient flight route in terms of, *e.g.*, travel time, or passenger “throughput” as already hinted at in the Motivation section.

More complex attribute sets can be encoded using elaborate particle motion behaviors. The air traffic dataset contains information about the average departure and arrival delays on each route. As illustrated in Figure 4.3, this information can be encoded by inserting two gates on the link to vary the speed of particles as they progress from origin to destination. Routes with no delay on either side will feature particles travelling at constant speed. Routes with departure delay will feature particles that start at a low speed, attaining their normal speed once they have travelled 20% of the link’s length. On routes with arrival delay, particles decelerate once they reach 80% of the link’s length.

4.3.1.2 Reducing Visual Complexity

Gates can also be used to reduce visual complexity in the diagram, for instance by removing some clutter. The node-link diagram from the previous example (Figure 4.3) features numerous crossing and overlapping edges, in part because of the constraints on the nodes, which are geolocated. Inspired by Becker *et al.*’s shortening lines [21], we used gates to fade particles away in the central portion of the links, as illustrated in Figure 4.4. Decreasing the particles’ opacity in this manner helps reduce visual clutter. Here, the motion of particles helps track a particular link, in part thanks to the Gestalt law of common fate that contributes to apprehending the particles that flow on that edge as one coherent group.

³<http://ilda.saclay.inria.fr/flownet>

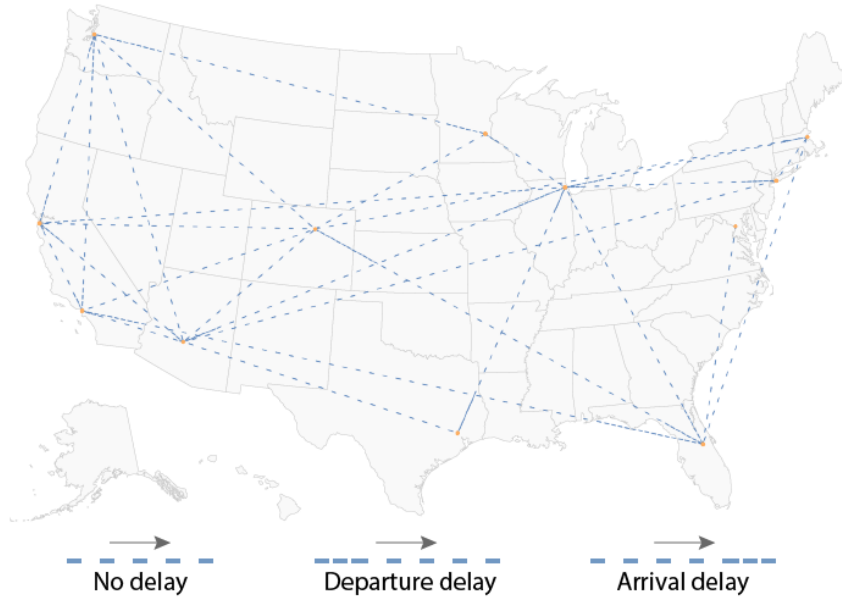


Figure 4.3: Particle speed mapped to departure and arrival delay.

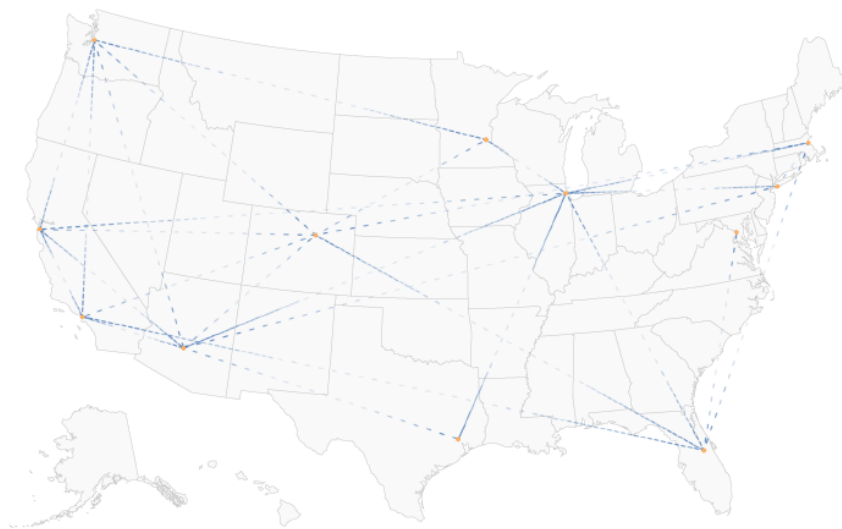


Figure 4.4: Same as in Figure 4.3, also reducing edge clutter by gradually fading particles out in the central portion of each edge.

4. USING MOTION AS AN ENCODING CHANNEL IN NODE-LINK DIAGRAMS

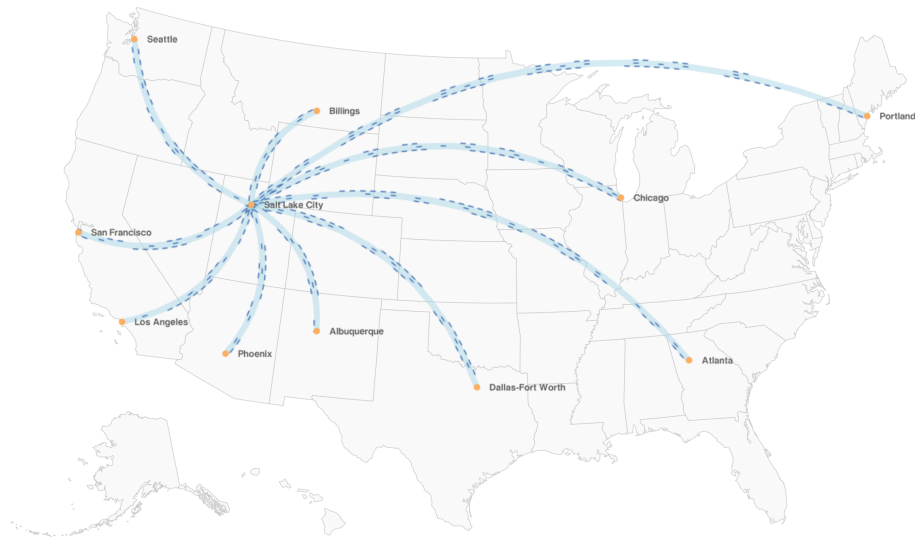


Figure 4.5: Using tracks to show different route types between airports.

4.3.1.3 Encoding a categorical attribute.

Categorical attributes are best encoded using variables that do not inherently convey an order, such as temporal pattern (which is akin to Morse code). For example, particles can be emitted according to different patterns depending on the route type (passenger, mail, freight). Difficulties will arise when an edge has more than one value for the same attribute. When using shape or color hue to encode categorical data, multiple values can be represented by, *e.g.*, merging the shapes, or creating a hatched pattern featuring all relevant color hues. Similarly, all relevant individual particle patterns can be juxtaposed to form a more elaborate pattern that is representative of this combination. However, as in the case of shape or color hue, this solution only works for a very small amount of values. It also requires a minimum edge length to be discernible, as the length of the pattern is equal to, and even slightly larger than, the sum of all individual patterns involved.

4.3.2 Encoding multiple attributes

Combining multiple attributes (one might want to also display, *e.g.*, the average number of passenger or cargo flights) can be achieved using motion variables in conjunction with other visual variables such as the particles' color or size. For combinations of numerical and categorical attributes, particle speed or frequency can encode the numerical attribute, while particle hue (color) encodes the categorical attribute. For combinations of numerical attributes, one can leverage visual variables such as size or opacity in conjunction with motion.

However, encoding multiple categories on a single edge can rapidly become visually complex. One possible solution is to use more than one track on an edge, each track being mapped to a different attribute. In Figure 4.5, the three tracks are associated with passengers, mail and freight, respectively. The category each track represents is visualized using a dual encoding: the track index (they are always ordered in the same

way), and one of three particle patterns. Indeed the track index is likely not sufficient, considering that links can have different orientations. Then, the average number of flights for each of the three categories on each route is encoded using the particles' speed, which can thus be different on each track.

4.4 Experiment 1: Encoding edge attributes with motion variables only

The above examples try to illustrate a set of representative encodings that use motion, but they say nothing about users' ability to perceive the values encoded in this manner. A few user studies of graph visualization techniques include one or more conditions that involve motion in the way animated edge textures do [80, 181]. However, to the best of our knowledge, there are no empirical data on the number and type of motion patterns of edge particles that users can discern. Based on the model presented earlier, we start to explore the design space defined by animated edge textures, and provide initial empirical data about their potential for encoding information.

We report on an experiment that aims at evaluating the encoding potential of each motion variable *individually*. This is an initial study, and more studies will be necessary to fully understand the interplay between motion variables as well as with other visual variables, as discussed later. In our experiment, participants were presented with node-link diagrams whose links featured different animated edge textures. They were asked to group edges based on the motion of particles.

Our primary factor is the type of motion variable ($Motion_Variable \in \{Pattern, Speed, Frequency\}$) that can take up to six different values (*i.e.*, participants have to identify up to six different groups of edges). *Pattern* is the pattern of particles flowing along the edge. *Speed* and *Frequency* are respectively the speed of particles and the frequency at which the source node emits the pattern. As mentioned above, our experimental design varies a single motion property at a time, keeping the other two properties set to their default value ($Pattern_0$, $Speed_0$ and $Frequency_0$). $Pattern_0$ is a single particle, *i.e.*, the simplest pattern. $Speed_0$ and $Frequency_0$ were chosen so as to ensure that there is always more than one occurrence of the pattern visible on the smallest link that the diagrams used in the experiment can feature.⁴

We used one of two different strategies for choosing the different values to test, depending on the property considered. In the case of *Pattern*, there is no natural order between different rhythms, which makes it a categorical property. Inspired by previous work on rhythmic interaction [56], we chose the six different particle patterns shown in Figure 4.1. In the case of *Speed* and *Frequency*, which are continuous properties, we had to ensure a minimal *difference threshold* between consecutive levels, above their *Just Noticeable Difference* (JND) [51] in order to ensure that participants could perceive them as different. We first explain how we chose this difference threshold, and then report on our experimental design and observations.

⁴In the *Pattern* condition, $Speed_0$ is 6 mm.s^{-1} , and $Frequency_0$ is 0.3 Hz. In the *Speed* condition, $Frequency_0$ is 1.2 Hz. In the *Frequency* condition, $Speed_0$ is 7.5 mm.s^{-1} . We also make nodes emit particles with a phase shift in order to prevent participants from discriminating frequencies among edges connected to a node simply by looking in its immediate vicinity.

4.4.1 Difference Threshold for *Speed* and *Frequency*

According to the Weber-Fechner law [51], the JND must be assessed as a constant proportion of the original stimulus I , meaning that detecting a change from an initially-high level requires a higher-amplitude change than detecting a change from an initially-low level: $JND = \Delta I/I$.

The notion of JND has been applied to various perceptual channels such as sound [171] and chromaticity [100]. Previous studies have considered perception of changes in speed or frequency, but those findings do not readily apply to the specific context of animated edge textures. For example, studies in experimental psychology have considered reaction time to a sudden change in velocity of a single stimulus [105]. This differs significantly from our context, where people must compare multiple stimuli (animated links in a diagram) that co-exist on screen. Huber *et al.* [81] have studied users' ability to identify a target group within a larger group, based on its difference in velocity or flickering frequency relative to an otherwise-uniform group. However, in their experiments, the elements were simple squares that filled the display area. In a node-link diagram, edges are not uniform elements paving the space. They are distant from one another, and vary in both direction and size.

We ran a pilot study to get an approximation (accurate-enough for our purposes) of the JND at different reference values for both the *Speed* and *Frequency* motion variables. Six participants were asked to tell whether or not they perceived a difference between a source edge and a target edge, which varied in their orientation. At the beginning of each trial, both edges were animated with particles flowing at the same *Speed* (resp. *Frequency*). Participants had to press a key repeatedly until they saw a difference. Then they had to say whether the difference was an increase or a decrease. For both motion variables, we observed difference thresholds between 0.13 and 0.26, with no significant effect of the difference in orientation between the source and the target edges on the perceived difference. We then chose thresholds significantly higher than the observed JNDs (0.7 for *Speed*, 0.5 for *Frequency*) based on pilot tests on actual node-link diagrams, accounting for the fact that comparing particle frequencies and speeds in a node-link diagram featuring dozens of nodes and edges is necessarily more difficult than the comparison of a single pair of edges in an otherwise empty display.

4.4.2 Encoding Edge Attributes

This experiment followed a between-subject design, with three independent groups of participants testing three different motion variables: *Pattern*, *Speed* and *Frequency*. Our experiment software is available for illustration and replication purposes.³

4.4.2.1 Task and Procedure

The experimental task consisted in grouping edges according to the values of the tested *Motion_Variable* (several links could have the same texture assigned to them). While such a task is definitely not a realistic visual graph analysis task, we believe it provides a relevant operationalization for the purpose of this visual perception experiment, as it requires participants to perform visual comparisons between edges not only to identify how many different values exist in the graph, but also to perform pairwise comparisons to identify which edges belong to the same group.

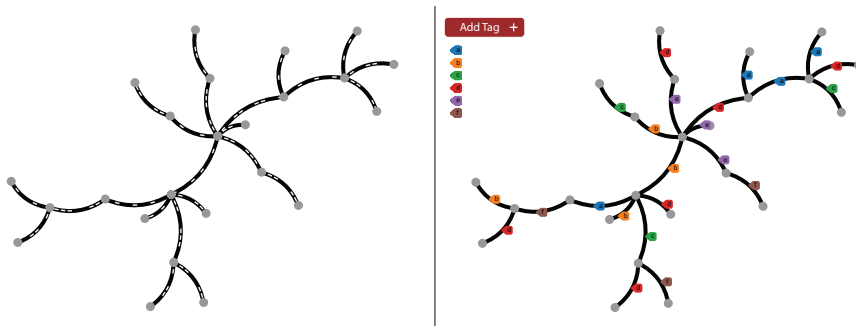


Figure 4.6: The second phase of our experimental task. The diagram on the left pane features animated edge textures, while the one on the right contains a copy of that diagram featuring static, solid edges. The right pane is interactive. It allows participants to create new tag types, which get added to the tag bar, and to tag individual edges. The background and edge colors have both been inverted in this screen capture: during the experiment, the background was black, and the edges were light gray.

As we aim at estimating the number of different values that each motion variable can encode, we want to find out whether users can group edges according to the different values of these properties present in the graph. Figure 4.6 shows a trial in the *Pattern* condition. The graph features six different *Pattern* values, meaning that participants should identify six groups of edges, provided they can actually discriminate all six values. A trial was divided into two phases.

In the first phase, participants had up to 45 seconds to tell how many different groups they perceived. They were encouraged to answer *as fast as they could*. They could press the space bar at any time before the timer expired. Either of these events (space bar pressed or the 45 seconds elapsed) made the graph disappear. Participants then had to input their answer in a text field. This task was an estimation task, as opposed to a subitization or counting task [67].

In the second phase, illustrated in Figure 4.6, participants had to actually identify these groups, creating the appropriate number of group tags and tagging each individual edge. Participants were instructed to be *as accurate as they could*. To avoid the tags interfering with the perception of the diagram, the display was split in two parts. The diagram on the left featured animated edge textures, while the one on the right contained a copy of that diagram featuring static, solid edges. Participants looked at the left-hand side diagram to identify groups, and used the right-hand side one to tag edges. They were able to create new tag types in the tag bar on-demand, using a button. Tagging an edge then entailed either dragging a tag from the bar and dropping it on that edge, or selecting a tag in the bar and clicking on the edge. The bar tag selection being persistent, the latter mechanism was especially useful to tag several edges in a row. Each edge could only hold one tag.

The operator insisted on the fact that it did not matter if the number of groups found in the second phase of a trial did not match that found in the first phase. The first phase rather aims at evaluating the potential of each *Motion_Variable* to convey a first impression, while the second step aims at capturing a more thorough analysis of the diagram (*i.e.*, where users have to tell whether two edges share the same animated texture or not in graphs featuring more or less diversity).

In addition to the *Motion_Variable* (*Pattern*, *Speed* and *Frequency*), our experiment also considered tasks of varying difficulty, which we operationalized using two factors.

First, we tested different graph sizes: *Graph_Size* \in {LOW, MEDIUM, HIGH}. The node-link diagram layout was computed using D3’s force-layout algorithm, with 8 nodes and 7 links for LOW, 16 nodes and 14 edges for MEDIUM, and 22 nodes and 21 links for HIGH. In order to avoid introducing too many factors in this first evaluation, we considered planar graphs only (the layouts did not contain any edge crossing).

Second, we varied the number of groups (*Group_Count*). Our purpose was to test whether increasing the number of different values of a motion variable diminishes users’ ability to discriminate them. Factor *Group_Count* \in {SMALL=2, MEDIUM=4, LARGE=6} corresponds to the number of different values that *Motion_Variable* could take. As already mentioned, we picked the six different rhythms shown in Figure 4.1, which we believed would be discriminable considering the range of edge sizes in our layouts (average: 3.18mm, min: 2.87mm, max: 3.45mm). Values of *Speed* and *Frequency* were computed as successive values right above the corresponding difference threshold (0.7 for *Speed*, 0.5 for *Frequency*) yielding, for each group count:

<i>Group_Count</i>	<i>Speed</i> values (mm.s ⁻¹)
SMALL	$G_1 = 3.3, G_2 = 5.6$
MEDIUM	$G_1 = 3.3, G_2 = 5.6, G_3 = 9.5, G_4 = 16.2$
LARGE	$G_1 = 3.3, G_2 = 5.6, G_3 = 9.5, G_4 = 16.2, G_5 = 27.5,$ $G_6 = 46.7$

<i>Group_Count</i>	<i>Frequency</i> values (Hz)
SMALL	$G_1 = 0.3, G_2 = 0.5$
MEDIUM	$G_1 = 0.3, G_2 = 0.5, G_3 = 0.7, G_4 = 1.0$
LARGE	$G_1 = 0.3, G_2 = 0.5, G_3 = 0.7, G_4 = 1.0, G_5 = 1.5,$ $G_6 = 2.3$

We chose to limit the maximum number of different values to 6 as a higher number of conditions would have resulted in an intractable experiment design. We ran pilot studies to identify an upper bound, which showed that differentiating 6 values was already challenging. In addition, going far beyond 6 values would have resulted in overly low-or-high speeds and frequencies, that would have been of little practical value.

Our three factors were tested following a mixed experiment design. To keep a reasonable experiment length (i.e., \sim an hour), *Motion_Variable* was tested as a between-subject factor, each participant seeing only one of the three conditions. *Group_Count* and *Graph_Size* were tested as within-subject factors, all participants seeing all *Group_Count* \times *Graph_Size* conditions. In order to get familiar with the task and the interactive tagging technique, participants started with 3 practice trials on small graphs featuring 2, 4, and 6 different groups. Then, each measured trial was replicated twice. In total, each participant had to complete 18 trials for analysis (3 *Group_Count* \times 3 *Graph_Size* \times 2 replications), presented in a random order. At the end of the experiment, participants filled in a short questionnaire asking them about the strategies that they used to identify the different groups of edges.

4.4.2.2 Participants & Apparatus

Thirty six volunteers (18 female), aged 21 to 47 year-old (average 30.8, median 29), participated in the experiment. We conducted the experiment on a PC Dell Precision

4.4. Experiment 1: Encoding edge attributes with motion variables only

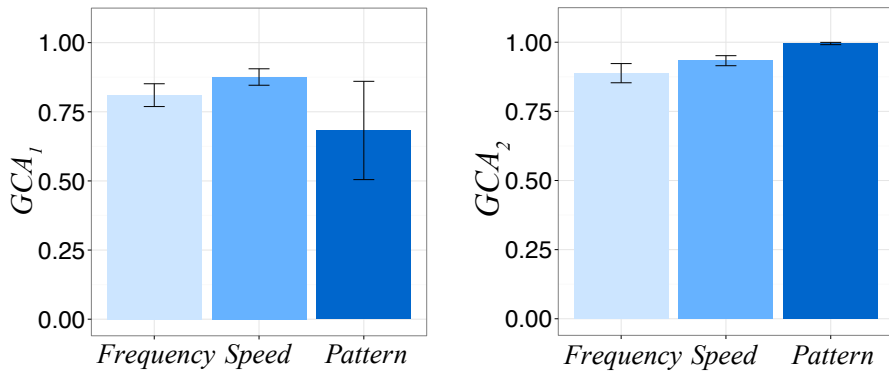


Figure 4.7: (Left) Accuracy in *estimating* the number of groups in the first phase; (Right) Accuracy in *counting* the number of different groups in the second phase. Error bars represent the 95% confidence interval.

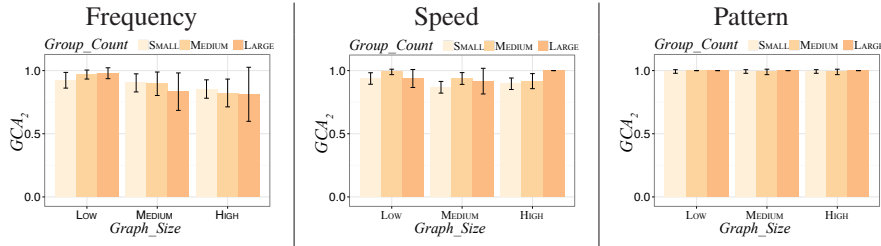


Figure 4.8: Average accuracy in second step (GCA_2) per $Graph_Size \times Group_Count$. Error bars represent the 95% confidence interval.

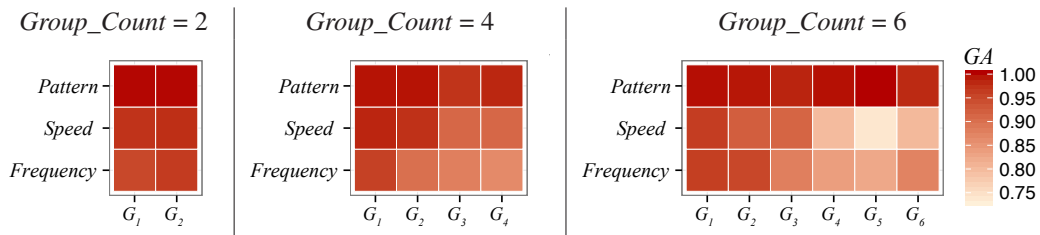


Figure 4.9: Confusion between groups per $Group_Count$ condition. In a matrix, a row corresponds to a $Motion_Variable$ condition, and a column to a group of edges. The color of a cell depends on the average accuracy of that group during the experiment (darker is better).

T5500, equipped with an Intel Xeon Quad Core processor, 8GB RAM, and an NVidia Quadro 2000 graphics card driving a 23" LCD FullHD 1080p monitor (1920x1080, 96 dpi). The experiment software was developed in JavaScript using node.js [172] and the animated edge texture library described earlier.

4.4.2.3 Data Collection & Data Analysis

Our experimental software collected the estimation of the number of groups that users reported in the first phase, GC_1 (which stands for Group Count in 1st phase). It also recorded the full set of mappings $\langle \text{tag}, \text{edge} \rangle$ of the 2nd phase, from which it derived GC_2 , the number of groups identified in that phase. We considered different measures for analyzing the collected data. We first computed the *Group Count Accuracy* (GCA) in both phases as follows:

$$GCA_1 = 1 - (|GC_1 - \text{Group_Count}|) / \text{Group_Count}, \text{ and}$$

$$GCA_2 = 1 - (|GC_2 - \text{Group_Count}|) / \text{Group_Count}.$$

However, GCA_2 alone is not sufficient to account for the grouping accuracy, as the number of tag groups can be accurate while having some edges misclassified. For each group, we thus also computed a measure of its accuracy using the Jaccard coefficient [84]. This coefficient estimates similarity between finite sample sets using the ratio between the size of the intersection and the size of the union of the sample sets. As we cannot know *a priori* what actual groups users were tagging, we try to associate each group G_i with the most similar tag group. The *Grouping Accuracy* of a group G_i , $GA(G_i)$, is thus the maximum of all Jaccard coefficients computed on each pair made of G_i and one of the groups T_j ($1 \leq j \leq GC_2$) that the participant has tagged:

$$GA(G_i) = \max_{1 \leq j \leq GC_2} \frac{\text{count}(T_j \cap G_i)}{\text{count}(T_j \cup G_i)} \quad 1 \leq i \leq \text{Group_Count}$$

4.4.2.4 Results

In the first phase, we observe an average accuracy GCA_1 of 0.81 for *Frequency*, 0.88 for *Speed* and 0.68 for *Pattern*. Participants seem to have more trouble estimating the number of groups when the discriminating motion variable is *Pattern*. However, an analysis of variance reveals that the difference between the three motion variables on GCA_1 is not significant ($p = 0.7$). In the second phase, the average accuracy GCA_2 increases to 0.88 for *Frequency*, 0.93 for *Speed* and 0.99 for *Pattern*. Here, the effect of *Motion_Variable* is significant on GCA_2 ($F_{2,33} = 3.5$, $p = 0.04$, $\eta_G^2 = 0.17$), with each pair of conditions significantly differing according to pairwise t-tests.⁵ Figure 4.7 illustrates these results. An interesting observation is that while *Pattern* performs relatively poorly as a means to get a quick *estimate* of the number of groups, it actually outperforms both other variables when the task is to more precisely *count* those groups. We tentatively explain this observation based on the fact that particle frequency and speed can be quickly perceived, while the identification of different patterns requires more cognitive processing. When trying to get an estimate under time pressure, different *Frequencies* or different *Speeds* will be easier to perceive, while when trying to get a precise count without such time pressure, *Pattern* will provide a cognitively more demanding, but more reliable, solution. Answers to the final questionnaire actually support this interpretation: participants reported experiencing difficulty memorizing patterns. On the opposite, in *Speed* and *Frequency* conditions, some of them reported using the

⁵We get the same results with or without Bonferroni correction.

4.4. Experiment 1: Encoding edge attributes with motion variables only

simple strategy of looking only at the spacing between two particles to discriminate different values.

Figure 4.8 provides a detailed view of how the different motion variables are robust against increasing difficulty. Even if an analysis of variance of $Graph_Size \times Group_Count$ on GCA_2 does not reveal any significant difference in any of the $Motion_Variable$ conditions, some edge properties seem to scale better to complex tasks (large graph, high number of groups) than others. *Pattern* and *Speed* seem to scale quite well with both $Graph_Size$ and $Group_Count$, while users' accuracy with *Frequency* appears to drop down on average as the graph becomes larger. However, our GCA_2 measure makes by definition the error cost inversely proportional to the number of groups (e.g., a misclassified edge impacts the accuracy three times more when $Group_Count=SMALL$ than when $Group_Count=LARGE$). We complement our error analysis with an absolute measure of errors, that is the number of misclassified edges NME . An analysis of variance of $Graph_Size \times Group_Count$ on NME revealed more contrasted differences. For *Pattern*, there is still no significant difference between conditions. But, when $Motion_Variable=Frequency$, $Graph_Size$ has a significant effect on NME ($F_{2,22} = 9$, $p = 0.001$, $\eta_G^2 = 0.27$), with NME significantly increasing with $Graph_Size$. Finally, when $Motion_Variable=Speed$, all $Graph_Size$, $Group_Count$ and $Graph_Size \times Group_Count$ interaction effects are significant on NME .⁶ Pairwise t-tests reveal that the effect of $Group_Count$ is significant only when $Graph_Size = HIGH$. For $SMALL$ - and $MEDIUM$ -sized graphs, the number of misclassified edges does not significantly increase with the number of groups.

Figure 4.9 helps understand where the confusion between groups comes from. In those matrices, a cell represents a group of edges G_i , and the color of that cell is proportional to the average *Grouping Accuracy* of that group $GA(G_i)$. The darker a cell, the most accurate a group is. We can observe that increasing the number of groups for *Frequency* and *Speed* makes the accuracy drop down. We can also observe that participants tend to make more confusions between high values of *Frequency* and *Speed*, the accuracy being lower starting from G_4 . It is important to remember that we increase $Group_Count$ by adding values at the end of the range, which entails that a group G_i , when it exists in two different $Group_Count$ conditions, is exactly the same in those two conditions.

In summary, *Pattern* seems to have good potential as a means to encode edge attributes. In this condition, participants in our experiment were able to identify the different edge groups almost perfectly, no matter the task difficulty. Participants also performed well with *Speed* and *Frequency*, but these properties seem to suffer more from an increase in the number of links. When *Frequency* or *Speed* are used to encode edge data attributes, users may experience some difficulty with large networks, and may experience difficulty discriminating between the higher frequencies and between the higher speeds.

4.4.3 Follow-Up Study

Answers to the questionnaire in the study reported above revealed that some participants had relied on the spacing between particles to compare different values of *Speeds* or

⁶ $Graph_Size$: $F_{2,22} = 10$, $p < 0.0001$, $\eta_G^2 = 0.27$; $Group_Count$: $F_{2,22} = 10$, $p < 0.0001$, $\eta_G^2 = 0.2$; and $Graph_Size \times Group_Count$: $F_{4,44} = 6$, $p < 0.0001$, $\eta_G^2 = 0.16$

Frequencies. Because *Speed* and *Frequency* can be manipulated independently in our model, modifying the value of one of these motions variables impacts the spacing between particles. Because this latter property can also be perceived on a static representation of the texture (e.g., on a snapshot of the graph), we also wanted to test what users can discriminate using motion dynamics only. We ran a follow-up study to test a fourth value for *Motion_Variable*, that we call *FASpeed*, for Frequency-Adjusted Speed. *FASpeed* adapts the frequency as speed varies so as to preserve a constant spacing between particles across the different *Speed* conditions.

As in the main experiment, we ran a pilot study to estimate the *Just Noticeable Difference* in a comparison task between a pair of edges. This informed the choice of consecutive values of *FASpeed*, computed using a threshold of 0.9:

<i>Group_Count</i>	<i>FASpeed</i> values (mm.s ⁻¹)
SMALL	[$G_1 = 3.75, G_2 = 7.125$]
MEDIUM	[$G_1 = 3.75, G_2 = 7.125, G_3 = 13, 5375, G_4 = 25, 725$]
LARGE	[$G_1 = 3.75, G_2 = 7.125, G_3 = 13, 5375, G_4 = 25, 725, G_5 = 48, 8775, G_6 = 93$]

4.4.3.1 Participants & Apparatus

Fifteen volunteers (3 female), aged 23 to 37 year-old (average 27.87, median 28), participated in this experiment. None of them was involved in the previous study. The experiment was run on a 2.7GHz Intel Core i5 Macbook Pro, equipped with 8GB RAM and an Intel Iris Graphics 6100 1536 Mo driving a 27" LCD monitor (2560x1440, 100 dpi).

4.4.3.2 Task & Procedure

Participants were exposed to the *Motion_Variable=FASpeed* condition only. As before, *Group_Count* and *Graph_Size* were tested using a within-subject design with two replications, with participants seeing a total of 18 trials for analysis. Trials were presented in a random order, after 3 practice trials. At the end of the experiment, participants filled in a short questionnaire about their strategies for grouping edges.

4.4.3.3 Results

We observe a similar trend to what we have observed in the previous experiment: accuracy increases between the 1st and the 2nd phases, with $GCA_1 = 0.82$ and $GCA_2 = 0.89$ on average. *Group_Count* does not have a significant effect on these two measures, suggesting that users can discriminate up to six different values. However, participants experienced more difficulty with large graphs than with small ones. The effect of *Graph_Size* is significant on both $GCA_1 (F_{2,28} = 8.86, p = 0.001, \eta_G^2 = 0.08)$ and $GCA_2 (F_{2,28} = 9.02, p = 0.001, \eta_G^2 = 0.11)$, with pairwise t-tests revealing that all *Graph_Size* conditions were significantly different from each other (LOW > MEDIUM > HIGH). Figure 4.10-(a) illustrates these results for GCA_2 .

The effect of *Graph_Size* on the total number of misclassified edges (*NME*) is larger ($F_{2,28} = 15.7, p = 0.00001, \eta_G^2 = 0.25$) than that of *Group_Count* ($F_{2,28} = 3.5, p = 0.04, \eta_G^2 = 0.08$). This is consistent with what we observed about *Speed* and *Frequency* in the previous experiment, where the number of misclassified edges grew with *Graph_Size*. However, as opposed to the *Speed* condition in the previous experiment, confusion

4.5. Experiment 2: Encoding edge attributes with both static and motion variables

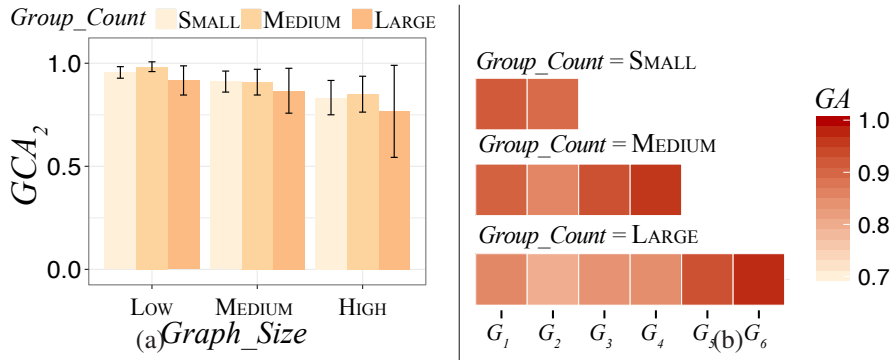


Figure 4.10: (a) Average accuracy in second step (GCA_2) per $Graph_Size \times Group_Count$. (b) Confusion between groups per $Group_Count$.

matrices (Figure 4.10-(b)) illustrate that confusions are more frequent between low values than high values of $FASpeed$.

In summary, participants were able to discriminate up to six different values based exclusively on the motion dynamics of flowing particles, with an accuracy of $\sim 90\%$. However, we also observe that the size of the graph impacts participants' accuracy. Large graphs involve comparisons between edges that can be far away from each other, and we observe that comparing two edges based on their dynamics is a difficult task when they do not both fall simultaneously in the center of visual attention. This is in line with participants' comments at the end of the experiment, where they reported having difficulty to "recall" the speed of the two edges that they wanted to compare when they were distant from each other.

4.5 Experiment 2: Encoding edge attributes with both static and motion variables

The two perception experiments reported in this section provide initial quantitative data about each motion variable studied in isolation. However, more empirical evidence needs to be gathered to fully understand the interplay between variables and comprehensively evaluate the potential of animated edge textures. In particular, it is not clear how motion variables interfere with variables that control the visual appearance of particles.

In this section, we report on a study that investigates the interplay between particle *speed*, two particle-color attributes (*luminance* and *chromaticity*), and two particle-size attributes (*length* and *width*). Results show that neither the *luminance*, nor the *chromaticity*, nor the *width* of particles interferes with their perceived *speed*. Only variations in their *length* interfere with the perception of their relative *speed* across edges. We discuss these findings and illustrate applications with simple examples of mappings that make effective use of combinations of encoding channels to represent multivariate edges.

We report on a series of experiments that assess users' ability to compare motion speed between two edges, when particles flowing along those edges also vary along a static visual variable. Each participant takes part in four experiments, presented in a random order, and performed in separate, non-consecutive sessions. Each experiment

4. USING MOTION AS AN ENCODING CHANNEL IN NODE-LINK DIAGRAMS

investigates the combination of speed and one of the following four visual variables (VV): *Length*, *Width*, *Luminance*, and *Chromaticity*. For example, for $VV=Width$, the experiment aims at answering the following practical question: do users perceive particles to be moving at the same speed when they are thin and when they are thicker?

4.5.1 Task

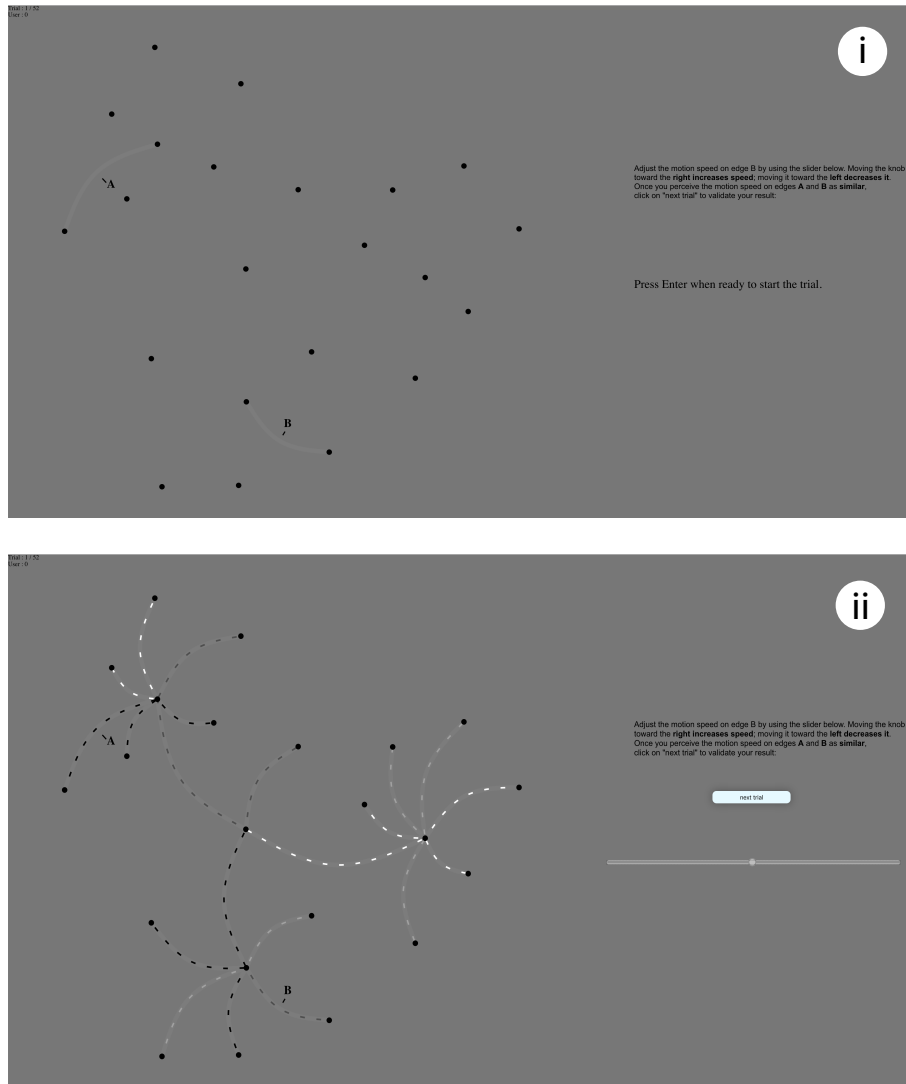


Figure 4.11: Experimental task. (i) The system first indicates to participants which two edges they will have to compare. (ii) When participants press the Enter key, the visualization gets animated. Participants have to adjust the speed of particles on edge B so that it matches that of particles on edge A, using the slider on the right.

Figure 4.11 illustrates the experimental task. Participants are instructed to focus on two edges only (indicated by A and B). They have to adjust the speed of particles on

4.5. Experiment 2: Encoding edge attributes with both static and motion variables

response edge B so as to make it match that of particles on *reference* edge A. Moving the slider knob toward the left decreases the speed of particles flowing along edge B. Moving the knob toward the right increases it. At the beginning of a trial, the slider knob is always positioned in the middle of the slider. The slider has 120 steps, each step corresponding to an increment (resp. decrement) in speed that is proportional to the speed of reference edge A ($slider_unit = \frac{speed(A)}{40}$). This design ensures that the target speed (*i.e.*, that of reference edge A) is contained in the slider's range, but that it does not always correspond to the same position of the knob relative to the slider across trials.

We generate eight planar graphs, which are drawn without edge crossings. The presentation of these graphs is counterbalanced across experimental conditions. Layouts consist of 20 to 28 links, and 21 to 29 nodes, and are generated using D3's force-directed algorithm [25]. Each layout meets the following requirements:

- nodes are distributed in a spatially uniform way;
- there is no pair of nodes that are too close to each other.

The resulting node-link diagrams have the following characteristics:

- average distance between connected nodes: 61mm (min=18, max=165, median=59);
- average distance of the two links to be compared: 147mm (min=105, max=194, median=148)⁷;
- average length of these links: 66mm (min=42, max=106, median=64)⁸;
- average relative difference in orientation between two links (modulo 180°): 56° (min=1°, max=172°, median=36°).

All eight layouts are available on the companion website, as well as the data collected during the study.⁹

The study of phenomena related to motion perception requires that we are very careful about the visual design of the experimental setup. In particular, we have to ensure that differences in luminance will not impact participants' perception of particles' color (*i.e.*, avoid phenomena such as "dark particles on a light background appear darker than they actually are"). Szafrir [166] showed that the minimal difference in luminance that is required in order not to impact the perception of color depends on the size of the visual mark considered. Based on her recommendations and on the minimal size that particles can have in our series of experiments, we computed a minimal difference of luminance of 11%.¹⁰ A background color of (50,0,0)_{CIELAB} and link

⁷Distance between links is computed as the minimum distance among all four pairs of endpoints between the two links.

⁸Calculation of link length ignores the link's curvature.

⁹To keep the submission anonymous, the companion website is provided only as supplemental material for now.

¹⁰We use formula (8) in [166] considering that particles are small elongated marks that can have various orientations.

4. USING MOTION AS AN ENCODING CHANNEL IN NODE-LINK DIAGRAMS

	speed of edge A ($SPEED_A$)	initial speed of edge B ($SPEED_{B/A}=Lower$)	initial speed of edge B ($SPEED_{B/A}=Higher$)
(<i>Low</i>)	5.36 mm.s ⁻¹	1.55 mm.s ⁻¹	10.18 mm.s ⁻¹
(<i>Medium</i>)	19.35 mm.s ⁻¹	10.18 mm.s ⁻¹	36.76 mm.s ⁻¹
(<i>High</i>)	36.76 mm.s ⁻¹	19.35 mm.s ⁻¹	69.94 mm.s ⁻¹

Table 4.1: Tested speeds in the experiment, in millimeters per second. The first column corresponds to the speed of *reference* edge A, which remains fixed throughout a trial. The second (resp. third) column corresponds to the initial speed of *response* edge B in condition $SPEED_{B/A}=Lower$ (resp. $SPEED_{B/A}=Higher$).






color of $(49,0,0)_{CIELAB}$ ensure that particles always have a minimal 16% difference in luminance with both the background and the link itself, in all tested conditions.






Each of the four experiments follows a within-subject design considering four factors:






- $SPEED_A$: the speed of *reference* edge A, $SPEED_A \in \{Low, Medium, High\}$;
- $SPEED_{B/A}$: the initial speed of *response* edge B relative to that of *reference* edge A, $SPEED_{B/A} \in \{Lower, Higher\}$;
- Δ_{VV} : the difference in value between the *reference* and *response* edges for the considered visual variable (*Length*, *Width*, *Luminance* or *Chromaticity*). The difference can be small, medium, large or there can be no difference ($\Delta_{VV} \in \{Same, Small, Medium, Large\}$). Figure 4.12 details the exact values corresponding to these different levels, per visual variable. Factor Δ_{VV} is handled differently in the case of *Chromaticity* as it is two-dimensional, compared to *Length*, *Width* and *Luminance*, which are one-dimensional. For *Chromaticity*, we consider the four colors that correspond to the extrema of the two axes defining the chromaticity space in the CIELAB model (at a constant luminance level of 75), and test the following four difference cases: no difference (*Same*), and difference with one of the three other colors.

Perception studies with animated gratings have shown that visual variables can alter perceived velocity to a different extent depending on how fast the grating moves (e.g., [55, 187]). They have also shown that the relative speed of another ‘modifier’ grating can affect the perceived velocity of the grating of interest [163]. In our experiment, factors $SPEED_A$ and $SPEED_{B/A}$ are introduced to account for these potential effects in the context of animated edge textures. To limit the duration of the experiment, we chose a sample of three values (*Low*, *Medium* and *High*) for the speed of *reference* edge A out of the six that were tested in [135]. Initial speed for *response* edge B was set relative to that of edge A in order to make the difference between edges sufficient to be perceivable without significant effort. Table 4.1 details the actual speeds we test in the different $SPEED_A \times SPEED_{B/A}$ conditions. According to recommendations from [135], the emission frequency of particles is adjusted depending on their speed, in order to preserve a constant spacing between them.

4.5. Experiment 2: Encoding edge attributes with both static and motion variables

Possible levels for VV in the <i>Length</i> experiment	
reference edge A	 $Length = 2.82\text{mm}; Width = 0.94\text{mm}; color_{CIELAB} = (L=0,0,0)$
response edge B	$Length \in \{v_1, v_2, v_3, v_4\}; Width = 0.94\text{mm}; color_{CIELAB} = (L=0,0,0)$
	 $v_1 = 2.82\text{mm}$  $v_2 = 4.935\text{mm}$  $v_3 = 8.64\text{mm}$  $v_4 = 12.95\text{mm}$ $\Delta = 0$ (<i>Same</i>) $\Delta = \text{Small}$ $\Delta = \text{Medium}$ $\Delta = \text{Large}$

Possible levels for VV in the <i>Width</i> experiment	
reference edge A	 $Length = 2.82\text{mm}; Width = 0.94\text{mm}; color_{CIELAB} = (L=0,0,0)$
response edge B	$Length = 2.82\text{mm}; Width \in \{v_1, v_2, v_3, v_4\}; color_{CIELAB} = (L=0,0,0)$
	 $v_1 = 0.94\text{mm}$  $v_2 = 1.55\text{mm}$  $v_3 = 2.56\text{mm}$  $v_4 = 4.22\text{mm}$ $\Delta = 0$ (<i>Same</i>) $\Delta = \text{Small}$ $\Delta = \text{Medium}$ $\Delta = \text{Large}$

Possible levels for VV in the <i>Luminance</i> experiment	
reference edge A	 $Length = 2.82\text{mm}; Width = 0.94\text{mm}; color_{CIELAB} = (L=0,0,0)$
response edge B	$Length = 2.82\text{mm}; Width = 0.94\text{mm}; color_{CIELAB} = (L \in \{v_1, v_2, v_3, v_4\}, 0, 0)$
	 $v_1 = 0\%$  $v_2 = 33\%$  $v_3 = 66\%$  $v_4 = 100\%$ $\Delta = 0$ (<i>Same</i>) $\Delta = \text{Small}$ $\Delta = \text{Medium}$ $\Delta = \text{Large}$






Possible levels for VV in the <i>Chromaticity</i> experiment	
reference edge A	 $Length = 2.82\text{mm}; Width = 0.94\text{mm}; color_{CIELAB} = (75,0,-128)$
response edge B	$Length = 2.82\text{mm}; Width = 0.94\text{mm}; color_{CIELAB} \in \{v_1, v_2, v_3, v_4\}$
	 $v_1 = (75,0,-128)$  $v_2 = (75,-128,0)$  $v_3 = (75,128,0)$  $v_4 = (75,0,128)$ $\Delta = 0$ (<i>Same</i>)

Figure 4.12: Possible values for all four visual variables (VV): *Length*, *Width*, *Luminance*, *Chromaticity*.

After a series of four practice trials (corresponding to a sample of possible conditions), each participant completes two blocks. Each block contains 48 trials, corresponding to the 24 ($3 \times 2 \times 4$) conditions, repeated twice. Conditions $SPEED_A \times \Delta_{VV}$ are presented in a random order. For each $SPEED_A \times \Delta_{VV}$ condition, the two conditions $SPEED_{B/A} = \text{Lower}$ and $SPEED_{B/A} = \text{Higher}$ are presented in series, but in a random order across conditions.

In each of the four experiments, motion variable *speed* is combined with a different visual variable *VV*: *Length*, *Width*, *Luminance*, *Chromaticity*. We have computed fifteen random orders of the four experiments to counterbalance their presentation across our fifteen participants.

4.5.2 Participants & Apparatus

Fifteen volunteers (4 female), aged 21 to 42 year-old (average 26, median 24), all with normal or corrected to normal vision, no color blindness, participated in the experiments. There was no remuneration involved. We conducted the experiments on a PC Dell Precision 5520, equipped with an Intel core i7-7820HQ processor (3.9GHz), 16GB RAM, and an NVidia Quadro M1200 graphics card (4GB), driving an 27" DELL U2715H external monitor (2560 x 1440 QHD, 109 ppi). The monitor features a 16:9 ratio and a luminosity of 350cd.m². Contrast ratio is 1000:1 (native), and 2000000:1 (dynamic). Participants are seated at a distance of 0.6m from the screen.

4.5.3 Results

The main measure of the experiment is Δ_{speed} : the absolute difference in speed between *response* edge B and *reference* edge A at the end of the trial. The lower this difference, the better participants are at estimating speed regardless of variations along another visual variable. In other words, Δ_{speed} is a measure of how much a visual variable (*Length*, *Width*, *Luminance* or *Chromaticity*) interferes with the speed motion variable.

As all participants took part in each of the four conditions and experienced varying presentation orders, data collected across the four experiments can be handled as a single experiment with four factors (*VV*, *SPEED_A*, *SPEED_{B/A}*, Δ_{VV}), whose design can be summarized as follows:

	15	users
×	4	levels of <i>VV</i>
×	2	blocks
×	3	levels of <i>SPEED_A</i>
×	4	levels of Δ_{VV}
×	2	levels of <i>SPEED_{B/A}</i>
=	2880	trials in total

A repeated measure ANOVA of the four factors on Δ_{speed} reveals a main effect for each of them.

We start our analysis with factor *SPEED_{B/A}* as it has a main effect ($F_{1,14} = 10.9$, $p < 0.0001$, $\eta_G^2 = 0.03$), but no interaction effect with other factors ($p > 0.05$). Participants were slightly less precise when they had to decrease the speed of response edge B to reach that of edge A ($\Delta_{speed} = 4.1 \text{ mm.s}^{-1}$ when *SPEED_{B/A} = Higher*) than when they had to increase it ($\Delta_{speed} = 3.1 \text{ mm.s}^{-1}$ when *SPEED_{B/A} = Lower*). This difference might be due to the fact that our ability to perceive changes is not linear. Actually, according to the Weber-Fechner law [51], detecting a change from an initially-high level requires a higher-amplitude change than detecting a change from an initially-low level. However, the effect of *SPEED_{B/A}* is rather small and, as it has no or negligible interaction with other factors, we ignore it for the rest of our analyses.

4.5. Experiment 2: Encoding edge attributes with both static and motion variables

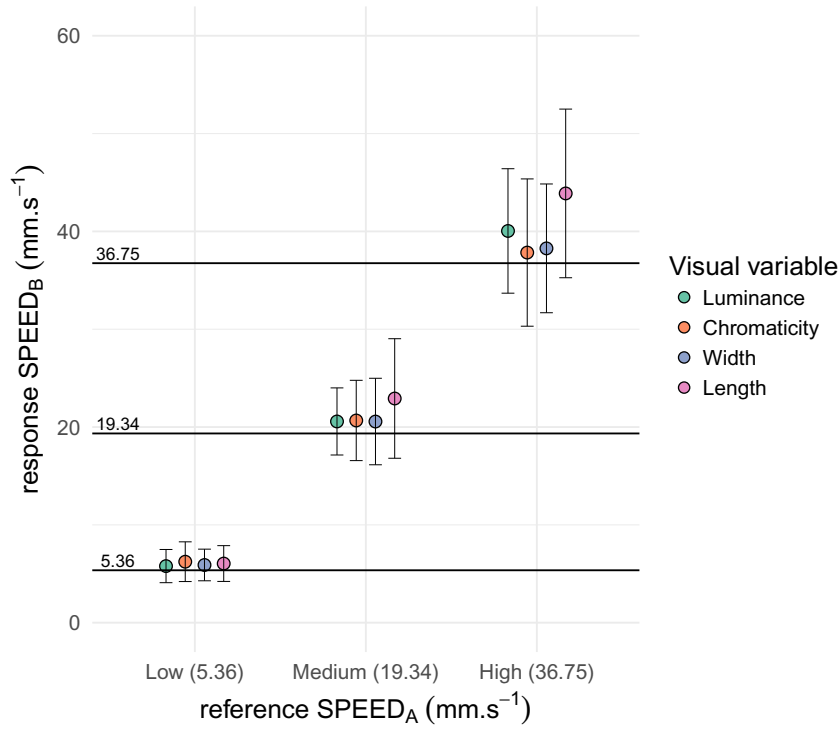


Figure 4.13: Average final speed of response edge B per $SPEED_A \times VV$ condition. Error bars represent the 95% confidence interval.

The remaining three factors all have a significant effect on Δ_{speed} , and they all interact with each other. First, factor $SPEED_A$ has a large effect on Δ_{speed} ($F_{2,28} = 51.8$, $p < 0.0001$, $\eta_G^2 = 0.31$), which supports the Weber-Fechner law mentioned above: participants were worse at estimating difference at high speeds than they were at low speeds. We then look at factor VV , which is our primary factor of interest, in order to observe whether visual variables of different nature have different levels of interference with the speed motion variable. An ANOVA reveals a significant effect of VV on Δ_{speed} ($F_{3,42} = 18$, $p < 0.0001$, $\eta_G^2 = 0.11$). As illustrated in Figure 4.13, participants seem to have more trouble in estimating particle speed when the particles vary in their length. Pairwise comparisons between VV conditions using paired t-tests show that only the *Length* condition is significantly different from all other VV conditions ($p < 0.0001$). The other three visual variables are not significantly different from each other. Figure 4.13 also illustrates the interaction effect between VV and $SPEED_A$ on Δ_{speed} ($F_{6,84} = 8$, $p < 0.0001$, $\eta_G^2 = 0.07$). Differences in luminance and length of particles have a higher impact on their perceived speed when particles move at a high speed. However, even in the $SPEED_A=High$ condition, only the effect of the *Length* visual variable is significant.

The last factor, Δ_{VV} , is the magnitude of the difference between the values of the considered visual variable on the reference and response edges. Its value must be interpreted relative to that of VV . We thus break down the rest of our analyses by VV condition, in order to better understand what happens under the different

4. USING MOTION AS AN ENCODING CHANNEL IN NODE-LINK DIAGRAMS

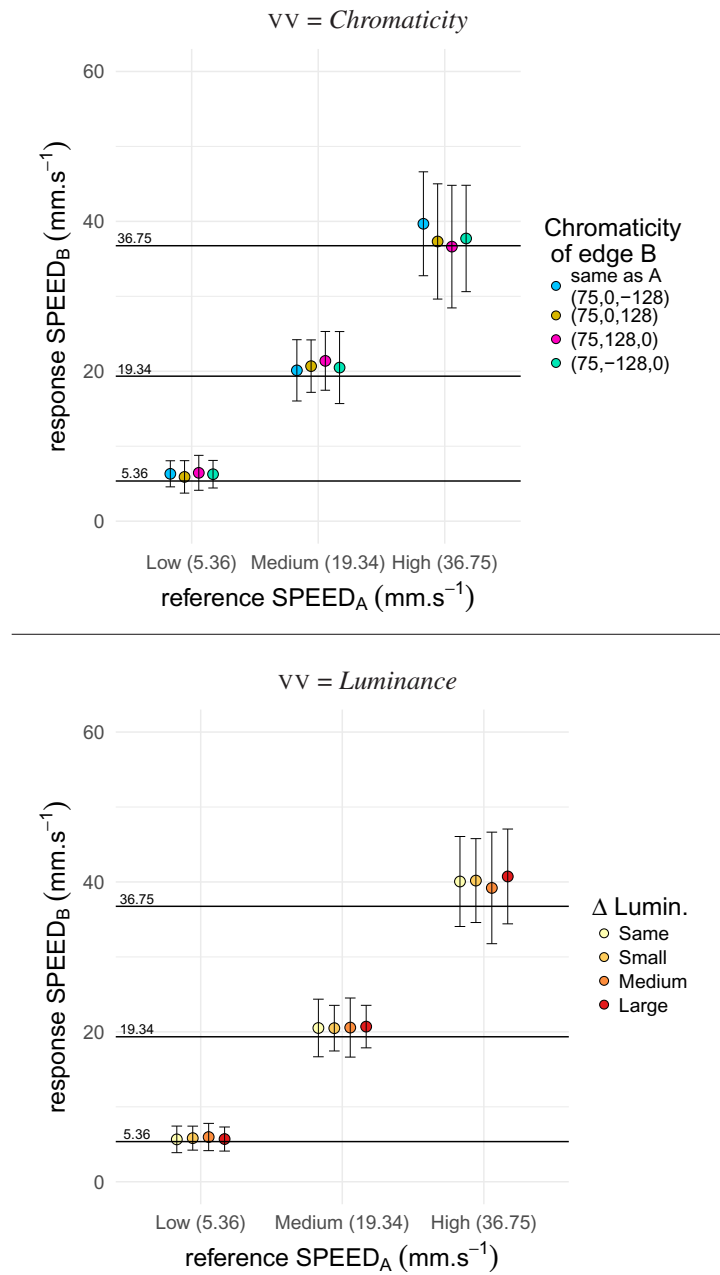


Figure 4.14: Average final speed of response edge B for each $SPEED_A \times \Delta_{VV}$ condition corresponding to color visual variables. Error bars represent the 95% confidence interval.

visual variable conditions. Figures 4.14 & 4.15 illustrate our results for each visual variable. For *Chromaticity*, *Luminance* and *Width*, only the effect of $SPEED_A$, already mentioned above, is significant on Δ_{speed} ($F_{2,28} = 22$, $p < 0.0001$, $\eta_G^2 = 0.41$, $F_{2,28} = 39$, $p < 0.0001$, $\eta_G^2 = 0.54$ and $F_{2,28} = 31$, $p < 0.0001$, $\eta_G^2 = 0.44$ respectively). Neither Δ_{VV}

4.5. Experiment 2: Encoding edge attributes with both static and motion variables

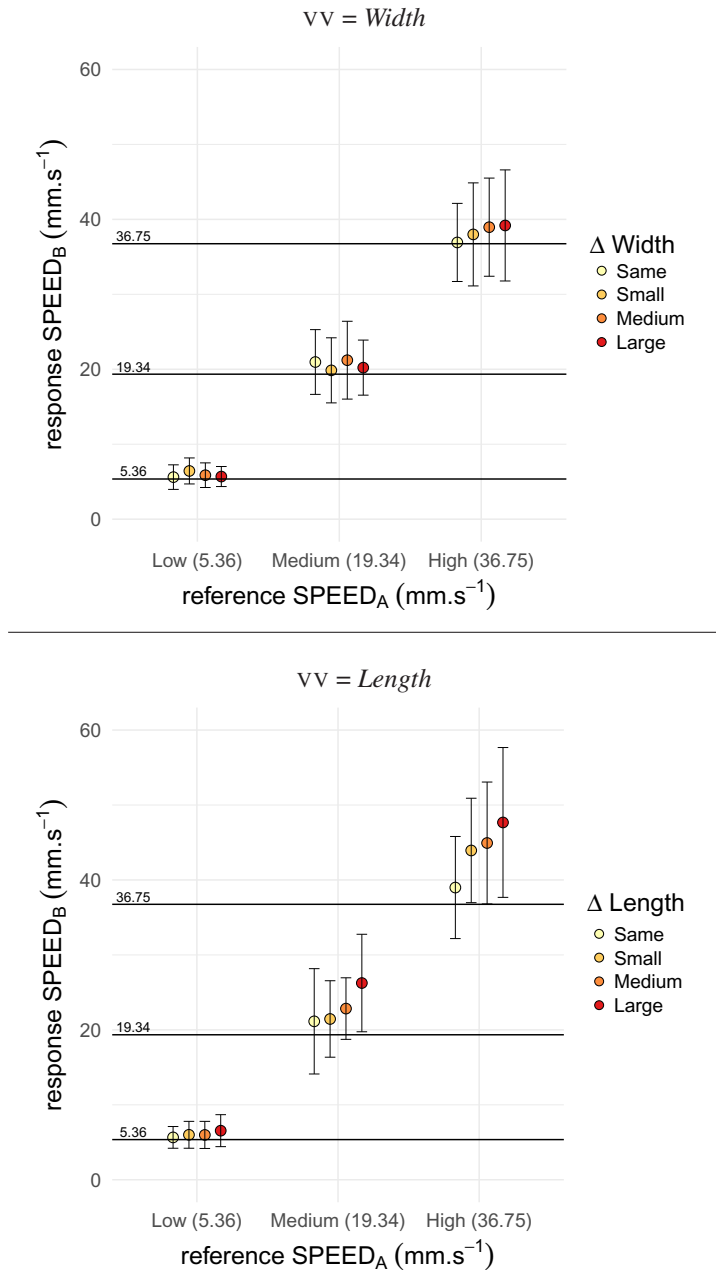


Figure 4.15: Average final speed of response edge B for each $SPEED_A \times \Delta_{VV}$ condition corresponding to size visual variables. Error bars represent the 95% confidence interval.

nor $SPEED_A \times \Delta_{VV}$ have a significant effect ($p > 0.05$). The situation is a bit different for *Length*. We observe three significant effects on Δ_{speed} : the effect of $SPEED_A$ as for other visual variables ($F_{2,28} = 43, p < 0.0001, \eta_G^2 = 0.54$), but also the effect of Δ_{VV} ($F_{3,42} = 13, p < 0.0001, \eta_G^2 = 0.23$), and of the interaction $SPEED_A \times \Delta_{VV}$ ($F_{6,84} = 7, p < 0.0001, \eta_G^2 = 0.12$). The accuracy in estimating speed gets worse on average with

larger differences in length, and this phenomenon gets amplified with the speed of particles. Pairwise comparisons between $\Delta_{VV} \times \text{SPEED}_A$ support this interpretation. For example, in the $\text{SPEED}_A=\text{Low}$ condition, only *Same* is different from *Medium* and *Large* ($p < 0.05$), while in $\text{SPEED}_A=\text{High}$ condition, almost all pairs of conditions significantly differ ($p < 0.05$). We tentatively attribute this to the size-speed illusion studied in experimental psychology (smaller objects appear to move faster than larger ones – see, e.g., [192]), although this remains speculation at this stage.

4.5.4 Summary of findings

In our experiment, the color of particles did not interfere with their perceived speed, suggesting that visualization designers can safely communicate two attributes on links using *Speed* and *Color* as the encoding channels. Regarding color, designers can make use of either chromaticity, which is typically useful to encode categorical attributes, or luminance, which is better suited to quantitative attributes. The perceived speed of particles was not affected by either of those variables in our empirical observations.

On the contrary, our study shows that variations in the length of particles' do have an impact on their perceived speed, and that interferences between speed and length get more important with large differences in length and at high speeds. This means that using *Speed* and *Size* in combination should be done with caution.

In particular, our results support that a *Speed+Width* encoding should be preferred over a *Speed+Length* one.

We illustrate these findings on two examples of possible combinations that do not cause interferences. The first combines *Speed* and *Color* to visually encode two edge attributes. The second combines *Speed* and *Size*. Animated versions of these node-link diagrams are available on the companion website.

Speed+Color Figure 4.16 shows an example of a node-link diagram that encodes edge attributes using a combination of particle speed and color. This node-link diagram shows data about air traffic in the USA, where nodes represent airports and links aerial routes connecting airports. The graph data are multi-variate, with multiple attributes for both airports and routes that call for the use of multiple encoding channels. A particle-based node-link diagram is especially well suited to represent air traffic, as it effectively conveys the notion of transit from one airport to the other. The color of particles encodes the main type of payload on a route (mail, freight or passengers). The speed of particles encodes the average number of planes traveling on that route.

The resulting visualization makes it possible to make comparisons per type of payload and across types of payloads. For example, the visualization shows that there are more planes with passengers traveling from Salt Lake City to Chicago than to Seattle, and that the number of planes carrying people to Chicago is higher than the number of planes carrying mail to San Francisco.

Speed+Size Figure 4.17 illustrates another example of an animated node-link diagram that shows Twitter activity of some politicians about two key topics: immigration and global warming. While the air traffic example above was using flowing particles to encode two attributes of different type (a categorical one and a quantitative one), this example makes use of particles to encode two quantitative attributes. Particle speed encodes the number of tweets mentioning a hashtag (shown as destination node) over the last year; particle width encodes the number of such tweets over the last month.

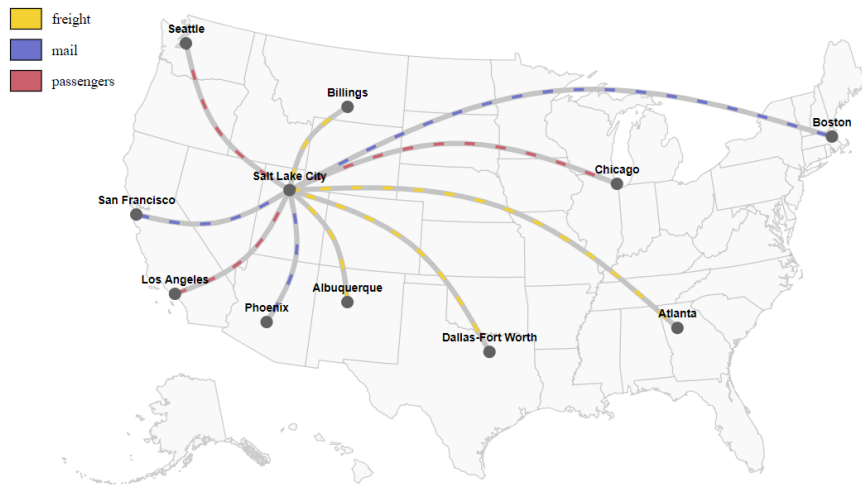


Figure 4.16: A node-link diagram representing air traffic from Salt Lake City to other airports in the USA. Particle chromaticity encodes the main category of payload: passengers in red, mail in blue and freight in yellow. Particle speed encodes the average number of planes on that route.

This design choice makes it possible to see actors' engagement on both a short- and a longer-term basis in the same node-link diagram. For example, if we consider hashtag "global warming", the speed of particles shows that Emmanuel Macron and Angela Merkel were more active over the entire year than Vladimir Putin and Donald Trump were, while the thick particles for Macron reveal that he was the most active over the last month.

4.6 Conclusion and Future Work

When designing node-link diagrams that feature animated particles, combining particle speed with other visual variables should be handled with caution. In this section, we observed that variations in chromaticity, luminance and width did not alter the perceived speed of particles, while variations in their length did. These empirical findings provide guidelines about which combinations between particle speed and these four visual variables can be effective. But these findings also suggest that the particle *pattern* should probably not be combined with particle speed. Indeed, particle patterns are obtained by introducing variable-length interspaces to delimit series of particles that form a pattern. An interspace between particles is conceptually close to a transparent particle, which should thus not feature variations in length so as not to interfere with perceived speed.

The effectiveness of animated edge textures in node-link diagrams might also be challenged in the context of graphs that are larger than the ones tested in these studies. Larger graphs will introduce a potentially larger distance between pairs of links that can make comparison based on motion difficult. Similarly, animated edge textures might be challenged with dense and non-planar graphs. A high density of links might

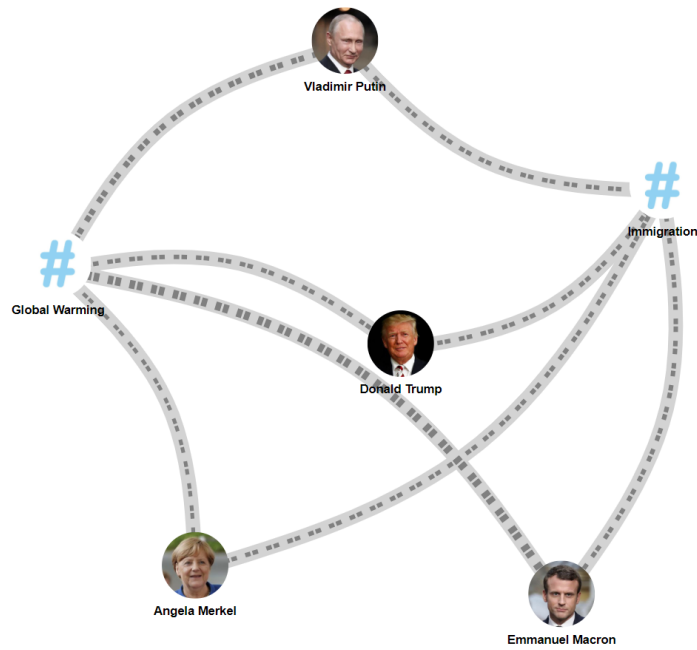


Figure 4.17: A node-link diagram representing activity of some politicians about key topics on the Twitter social network (*the data are fictional, generated randomly for illustration purposes only*). The width of particles encodes the number of tweets mentioning the destination hashtag (e.g., Global Warming) over the last month. The speed of particles encodes the number of such tweets over the last year.

introduce many motion ‘distractors’, while link crossings might impact users’ ability to follow flows of particles. Such effects remain to be studied in order to assess how the use of animated particle flows, as encoding channels, scale with graph size and complexity. Finally, our results provide guidelines about combining speed with static visual variables when considered in isolation. Combinations of multiple static variables in the context of animated edge textures should be studied next. For example, luminance and chromaticity taken together might interfere with the perception of motion [187].

Conclusion

During the last decade, the **amount** of data has constantly increased. Such data can come from several sources such as smartphones, audio recorders, cameras, or other sensors, or e-commerce, open data and can have **varying** structure. While computers can help us process these data, human judgment and domain expertise is what turns it into actual knowledge. However, **making sense** of this increasing amount of diverse data requires new visualization and interaction techniques. This thesis contributes such techniques to facilitate data exploration and presentation in relation with sense-making activities.

We present 4 contributions, with two of them more focused on data exploration and the two others more on data presentation. First, we presented ActiveInk that enables the natural use of pen for active reading of diverse data visualization. Then, we introduced SpaceInk, a design space of pen & touch techniques that make space for in-context annotations during active-reading of text documents. We presented Graphies, a prototype environment for expressive node-link diagram authoring that supports a flexible iterative design process. Finally, we introduced a set of motion variables to extend the set of encoding variables that are available to designers to represent data attributes associated with edges in multi-variate networks. We give a short summary of each contribution below, and then open directions for future work.

4.6.1 Summary

In the first part of this thesis, we focused on interactive systems and interaction techniques to support data exploration with the aim of making users able to externalize their thoughts more fluidly with a digital pen. ActiveInk aims at better supporting the sense-making process by using digital ink to transition between externalization and exploration of diverse visualizations. Through a qualitative study with eight participants, we contributed observations of active reading behaviors during data exploration and design principles to support sense-making. Second, SpaceInk is a design space of interaction techniques that can make space for pen writing in reflowable document. SpaceInk techniques make it possible to tightly integrate handwritten annotations with their scope in the document content. We identified representative techniques in this design space, spanning both new ones and existing ones. We evaluated them in a user study, whose results informed the design of a prototype system. This final prototype lets users concentrate on capturing fleeting thoughts, streamlining the overall annotation process by enabling the fluid interleaving of space-making gestures with freeform ink.

In the second part, we focused on techniques to visually represent insights and answers to questions that arise during sense-making activities. We focused here on node-link diagrams as TKM analysts work mostly with multivariate networks. We investigated how to enable a flexible iterative design process when authoring node-link diagrams for multivariate networks. We demonstrated our approach with one system, and a set of motion variables. First, we present Graphies, a system that better supports creativity in the design workflow of node-link diagrams by making the design process flexible and iterative. We reported on a study where participants had to reproduce several expressive designs, and create their own designs as well. Then, we presented Animated Edge Textures, a set of motion variables that extend encoding possibilities available to designers to represent data attributes associated with edges in multi-variate networks. In two users studies, we evaluate their efficiency from a perception perspective.

4.6.2 Perspectives

During this thesis, we have mainly focused on the data exploration process to support sense-making activities, and on the presentation process to visually represent insights and answers to questions gained from such sense-making activities. However, there are still many research problems to address in both themes, as well as at their intersection. We have also focused our work on regular devices that support pen touch interaction, such as the Microsoft Surface Book that we have used in almost all of our studies. We believe that we could do better by taking advantage of multiple devices. In particular, specific sensing capabilities can support novel ways of exploring and consuming data. I present my future direction in the following section along these two axes: 1) using novel modalities for sense-making, and 2) removing the hard frontier between data exploration and data presentation.

4.6.2.1 Optimizing interaction techniques based on hardware capabilities

During the last 20 years, we have witnessed a phenomenal evolution of digital devices. Those devices can be extremely large such as the Wilder Wall [19] or the Microsoft surface Hub, or smaller such as the Apple iPad or the Microsoft Surface. Most of them support touch input but some of them also support more: for example pen touch technology has also become mainstream nowadays. Those devices offer new opportunities for people to explore and consume data, and afford workspaces that provide new interaction paradigms.

Researchers have started to explore how to foster unification and interoperability between those devices during data analysis and exploration activities [12]. However, interaction is usually not optimized for each of these devices, with a UI very similar from one device to the other, without taking advantage of a potentially richer input. For instance, when using Microsoft Word or Powerpoint and switching between a desktop computer and a tactile device, the same interaction design is applied, without taking advantage of the expressiveness of touch interaction: a click is transformed into a tap, a double-click into a double tap, *etc.* Such applications could take into account hardware capabilities that each device offers to make more features for data exploration and consumption available.

When a pen becomes available, it makes it possible to rely on hand writing for navigating in documents as well as for creating or modifying content. For example, inking could be used for a faster navigation: users could directly ink short addresses in the browser URL bar, or write the page number they would like to navigate to in a PDF document. Such examples would extend the approach that we develop in ActiveInk, where ink does more than just writing. Annotations can also be used as a reminder: users could ink in the scroll bar, as a reminder of information they have explored in a specific part of a document. Integrated together, such functionalities could help and encourage people to optimize the use of their device capabilities. Future work needs to investigate how to effectively transpose interaction techniques between devices, going beyond the basic point-and-select inherited from mouse-based devices.

When working with large touch screen devices, such as the Microsoft Studio, users orient it at different angles, either horizontal, vertical or oblique (Figure 4.18). While the horizontal mode is preferred for performing bi-manual interaction using pen touch for authoring documents, creating content or annotating, the vertical mode is preferred



Figure 4.18: Several tilt orientation when working with the Surface Studio

for reading or showing information to other people. Users orient their devices with different angles depending on the tasks they are performing, and the level of privacy they need, either public for reading and consuming data, either more private for writing annotations, or reviewing a document.

Several research projects have already used tilt input, but as a controller to navigate through information. Roy et al [140], used the orientation of a smartphone to design pointing techniques, allowing users to interact with virtual content displayed over the real world, in active situations like standing and walking. In TiltType and TiltText [186, 123], users can enter a text by pressing on specific buttons (on cell-phone keypad) and tilting the device in the correct position to validate the letter. In such projects, tilt is used as an explicit input channel, ignoring users' natural way of changing device orientation. The natural tilt angle users adopt with their devices when working with applications, has never been taken in account. For instance, after having reviewed a document in a horizontal position, when orienting the surface vertically, annotations could fade out or go in the margin to support an easy reading. Further work should investigate in which situations the use of tilt is relevant, and how to design effective techniques that take into account device orientation.

4.6.2.2 Exploration vs Presentation

In the literature about data visualization, most tools are either presentation or exploration oriented. For instance, some tools have been designed for presentation purposes only, such as Powerpoint, or DataToon [89], while others aim to explore data such as Pajek [120], Excel or Tulip [8]. The relatively strict boundary between those processes does not yet allow a seamless transition between data exploration and presentation. Users who want to build a presentation as well as explore their data need to go back and forth between several applications, such as Excel and Powerpoint. However, both exploration and presentation are intrinsically connected and should be designed according to one another. In fact, users who want to find relevant information and explore data usually want to present their findings afterwards or record a clear presentation for personal use. Also, during an exploration process, it can help to work with temporary representations to make information more readable and better support the overall reasoning process about a higher-level question.

Graphies begins to unify presentation and exploration processes, with, e.g., a timeline that allows users to organize several visualizations in the history to ease the exploration process. In such presentation oriented tools, we could even further support exploration by including graph metrics such as centrality or clustering coefficient. In exploration oriented applications such as ActiveInk, presentation features could be added to support a narrative process, such as replaying the exploration process that led to a specific result.

Future work should investigate how systems could effectively support the transition between exploration and presentation in a smooth manner, and how to integrate both processes in a single workspace.

During our work, we have mostly investigated how people make sense of data with visual representations such as node-link diagrams, maps, or barcharts. Such representations are commonly used by people to understand, explore and present their data thanks to their effectiveness to convey simple information. However, users also work with more unstructured data such as images or videos. Those data formats raise new challenges and opportunities for people to extract more information from collections of various data. We have started, with SpaceInk, to show how we can support sense-making with text. We believe that similar techniques could be designed to support sense-making with images and videos. Further studies should investigate how people make sense of collection of videos and images, and how people extract insights from them.

Bibliography

- [1] James Abello, Frank van Ham, and Neeraj Krishnan. Ask-graphview: A large scale graph visualization system. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):669–676, 2006, doi:10.1109/TVCG.2006.120.
- [2] Eytan Adar. Guess: A language and interface for graph exploration. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pages 791–800. ACM, 2006, doi:10.1145/1124772.1124889.
- [3] Maneesh Agrawala and Michael Shilman. Dizi: A digital ink zooming interface for document annotation. In *Proceedings of the IFIP TC13 International Conference on Human-Computer Interaction*, INTERACT'05, pages 69–79. Springer-Verlag, 2005, doi:10.1007/11555261_9.
- [4] Christopher Andrews, Alex Endert, and Chris North. Space to think: Large high-resolution displays for sensemaking. In *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, pages 55–64. ACM, 2010, doi:10.1145/1753326.1753336.
- [5] Lisa Anthony and Jacob O. Wobbrock. \$n-protractor: A fast and accurate multistroke recognizer. In *Proceedings of Graphics Interface 2012*, GI '12, pages 117–120, Toronto, Ont., Canada, Canada, 2012. Canadian Information Processing Society.
- [6] Caroline Appert, Michel Beaudouin-Lafon, and Wendy E. Mackay. Context matters: Evaluating interaction techniques with the cis model. In Sally Fincher, Panos Markopoulos, David Moore, and Roy Ruddle, editors, *People and Computers XVIII —Design for Life*, pages 279–295, London, 2005. Springer, doi:10.1007/1-84628-062-1_18.
- [7] Caroline Appert and Shumin Zhai. Using strokes as command shortcuts: Cognitive benefits and toolkit support. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages 2289–2298. ACM, 2009, doi:10.1145/1518701.1519052.
- [8] David Auber. *Tulip —A Huge Graph Visualization Framework*, pages 105–126. Springer, 2004, doi:10.1007/978-3-642-18638-7_5.
- [9] B. Bach, N. Henry Riche, C. Hurter, K. Marriott, and T. Dwyer. Towards unambiguous edge bundling: Investigating confluent drawings for network visualization. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):541–550, Jan 2017, doi:10.1109/TVCG.2016.2598958.

BIBLIOGRAPHY

- [10] Benjamin Bach, Natalie Kerracher, Kyle Wm. Hall, Sheelagh Carpendale, Jessie Kennedy, and Nathalie Henry Riche. Telling stories about dynamic networks with graph comics. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 3670–3682, New York, NY, USA, 2016. ACM, doi:10.1145/2858036.2858387.
- [11] Benjamin Bach, Emmanuel Pietriga, and Jean-Daniel Fekete. Graphdiaries: Animated transitions and temporal navigation for dynamic networks. *IEEE Transactions on Visualization and Computer Graphics*, 20(5):740–754, 2014, doi:10.1109/TVCG.2013.254.
- [12] Sriram Karthik Badam, Andreas Mathisen, Roman Rädle, Clemens Nylandsted Klokmose, and Niklas Elmqvist. Vistrates: A component model for ubiquitous analytics. *IEEE Trans. Vis. Comput. Graph.*, 25(1):586–596, 2019, doi:10.1109/TVCG.2018.2865144.
- [13] David Barger and Tomer Moscovich. Reflowing digital ink annotations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '03, pages 385–393. ACM, 2003, doi:10.1145/642611.642678.
- [14] Josh Barnes and Piet Hut. A hierarchical $O(n \log n)$ force-calculation algorithm. *Nature*, 324:446 EP–, 12 1986.
- [15] Lyn Bartram. Perceptual and interpretative properties of motion for information visualization. In *Proceedings of the 1997 Workshop on New Paradigms in Information Visualization and Manipulation*, NPIV '97, pages 3–7, New York, NY, USA, 1997. ACM, doi:10.1145/275519.275520.
- [16] Lyn Bartram and Colin Ware. Filtering and brushing with motion. *Information Visualization*, 1(1):66–79, 2002, doi:10.1057/palgrave.ivs.9500005.
- [17] Lyn Bartram and Miao Yao. Animating causal overlays. In *Computer Graphics Forum*, volume 27, pages 751–758. Wiley Online Library, 2008, doi:10.1111/j.1467-8659.2008.01204.x.
- [18] Mathieu Bastian, Sebastien Heymann, Mathieu Jacomy, et al. Gephi: an open source software for exploring and manipulating networks. In *3rd International Conference on Weblogs and Social Media*, ICWSM, pages 361–362. AAAI, 2009, doi:10.13140/2.1.1341.1520.
- [19] M. Beaudouin-Lafon, S. Huot, M. Nancel, W. Mackay, E. Pietriga, R. Primet, J. Wagner, O. Chapuis, C. Pillias, J. Eagan, T. Gjerlufsen, and C. Klokmose. Multisurface interaction in the wild room. *Computer*, 45(4):48–56, April 2012, doi:10.1109/MC.2012.110.
- [20] Fabian Beck, Michael Burch, Stephan Diehl, and Daniel Weiskopf. The State of the Art in Visualizing Dynamic Graphs. In R. Borgo, R. Maciejewski, and I. Viola, editors, *EuroVis - STARS*. The Eurographics Association, 2014, doi:10.2312/eurovisstar.20141174.
- [21] R. A. Becker, S. G. Eick, and A. R. Wilks. Visualizing network data. *IEEE Transactions on Visualization and Computer Graphics*, 1(1):16–28, Mar 1995, doi:10.1109/2945.468391.

-
- [22] Anastasia Bezerianos, Fanny Chevalier, Pierre Dragicevic, Niklas Elmqvist, and Jean-Daniel Fekete. Graphdice: A system for exploring multivariate social networks. In *Proceedings of the 12th Eurographics / IEEE - VGTC Conference on Visualization*, EuroVis'10, pages 863–872. Eurographs Association, 2010, doi:10.1111/j.1467-8659.2009.01687.x.
- [23] Alex Bigelow, Steven Drucker, Danyel Fisher, and Miriah Meyer. Reflections on how designers design with data. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '14, pages 17–24. ACM, 2014, doi:10.1145/2598153.2598175.
- [24] Alex Bigelow, Steven Drucker, Danyel Fisher, and Miriah Meyer. Iterating between tools to create and edit visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):481–490, January 2017, doi:10.1109/TVCG.2016.2598609.
- [25] Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. D3 data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, 2011, doi:10.1109/TVCG.2011.185.
- [26] Liliana Bounegru, Tommaso Venturini, Jonathan Gray, and Mathieu Jacomy. Narrating networks: Exploring the affordances of networks as storytelling devices in journalism. *Digital Journalism*, 5(6):699–730, 2017, doi:10.1080/21670811.2016.1186497.
- [27] Peter Brandl, Clifton Forlines, Daniel Wigdor, Michael Haller, and Chia Shen. Combining and measuring the benefits of bimanual pen and direct-touch interaction on horizontal interfaces. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '08, pages 154–161, New York, NY, USA, 2008. ACM, doi:10.1145/1385569.1385595.
- [28] A. J. Bernheim Brush, David Barger, Anoop Gupta, and J. J. Cadiz. Robust annotation positioning in digital documents. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '01, pages 285–292. ACM, 2001, doi:10.1145/365024.365117.
- [29] J. J. Cadiz, Anop Gupta, and Jonathan Grudin. Using web annotations for asynchronous collaboration around documents. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, CSCW '00, pages 309–318. ACM, 2000, doi:10.1145/358916.359002.
- [30] Drini Cami, Fabrice Matulic, Richard G. Calland, Brian Vogel, and Daniel Vogel. Unimanual pen+touch input using variations of precision grip postures. In *Proceedings of the 31st Symposium on User Interface Software and Technology*, UIST '18, pages 825–837. ACM, 2018, doi:10.1145/3242587.3242652.
- [31] Bay-Wei Chang, Jock D. Mackinlay, Polle T. Zellweger, and Takeo Igarashi. A negotiation architecture for fluid documents. In *Proceedings of the Symposium on User Interface Software and Technology*, UIST '98, pages 123–132. ACM, 1998, doi:10.1145/288392.288585.
- [32] M. A. Chatti, T. Sodhi, M. Specht, R. Klamma, and R. Klemke. u-annotate: An application for user-driven freeform digital ink annotation of e-learning

BIBLIOGRAPHY

- content. In *IEEE International Conference on Advanced Learning Technologies (ICALT'06)*, pages 1039–1043, July 2006, doi:10.1109/ICALT.2006.1652624.
- [33] Wei Chen, Fangzhou Guo, and Fei-Yue Wang. A survey of traffic data visualization. *IEEE Transactions on Intelligent Transportation Systems*, 16(6):2970–2984, 2015, doi:10.1109/TITS.2015.2436897.
- [34] Mauro Cherubini, Gina Venolia, Rob DeLine, and Andrew J. Ko. Let's go to the whiteboard: How and why software developers use drawings. In *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, pages 557–566. ACM, 2007, doi:10.1145/1240624.1240714.
- [35] Fanny Chevalier, Nathalie Henry Riche, Catherine Plaisant, Amira Chalbi, and Christophe Hurter. Animations 25 years later: New roles and opportunities. In *Proceedings of the International Working Conference on Advanced Visual Interfaces, AVI '16*, pages 280–287, New York, NY, USA, 2016. ACM, doi:10.1145/2909132.2909255.
- [36] Andy Cockburn, Carl Gutwin, and Jason Alexander. Faster document navigation with space-filling thumbnails. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '06*, pages 1–10. ACM, 2006, doi:10.1145/1124772.1124774.
- [37] Christopher Collins, Gerald Penn, and Sheelagh Carpendale. Bubble sets: Revealing set relations with isocontours over existing visualizations. *IEEE Trans. on Visualization and Computer Graphics*, 15(6):1009–1016, 2009, doi:10.1109/TVCG.2009.122.
- [38] Owen Cornec and Romain Vuillemot. Visualizing the scale of world economies. In *VisWeek 2015 Electronic Conference Proceedings - Poster*, 2015.
- [39] Tarik Crnovrsanin, Isaac Liao, Yingcai Wuy, and Kwan-Liu Ma. Visual recommendations for network navigation. In *Proceedings of the 13th Eurographics / IEEE - VGTC Conference on Visualization, EuroVis'11*, pages 1081–1090. Eurographics Association, 2011, doi:10.1111/j.1467-8659.2011.01957.x.
- [40] Andrew Crotty, Alex Galakatos, Emanuel Zraggen, Carsten Binnig, and Tim Kraska. Vizdom: Interactive analytics through pen and touch. *Proc. of the VLDB Endowment*, 8(12):2024–2027, 2015, doi:10.14778/2824032.2824127.
- [41] Pedro Cruz. Wrongfully right: applications of semantic figurative metaphors in information visualization. In *IEEE VIS Arts Program, VISAP*, pages 14–21, 2015.
- [42] Jos Dirksen. *Learning Three.js – the JavaScript 3D Library for WebGL*. Packt Publishing Ltd, 2015.
- [43] Jon Driver and Peter McLeod. Reversing visual search asymmetries with conjunctions of movement and orientation. *Journal of Experimental Psychology: Human Perception and Performance*, 18(1):22–33, 1992, doi:10.1037/0096-1523.18.1.22.

-
- [44] Jon Driver, Peter McLeod, and Zoltan Dienes. Motion coherence and conjunction search: Implications for guided search theory. *Perception & Psychophysics*, 51(1):79–85, 1992, doi:10.3758/BF03205076.
- [45] Steven M. Drucker, Danyel Fisher, Ramik Sadana, Jessica Herron, and m.c. schraefel. Touchviz: A case study comparing two interfaces for data analytics on tablets. In *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, pages 2301–2310. ACM, 2013, doi:10.1145/2470654.2481318.
- [46] Cody Dunne, Nathalie Henry Riche, Bongshin Lee, Ronald Metoyer, and George Robertson. Graphtrail: Analyzing large multivariate, heterogeneous networks while supporting exploration history. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 1663–1672. ACM, 2012, doi:10.1145/2207676.2208293.
- [47] Tim Dwyer. cola.js. <http://marvl.infotech.monash.edu/webcola/> - Last accessed 2017-09-08, 2013.
- [48] John Ellson, Emden R. Gansner, Eleftherios Koutsofios, Stephen C. North, and Gordon Woodhull. Graphviz and dynagraph – static and dynamic graph drawing tools. In *Graph Drawing Software*, pages 127–148. Springer-Verlag, 2003, doi:10.1007/978-3-642-18638-7_6.
- [49] Niklas Elmqvist, Nathalie Henry, Yann Riche, and Jean-Daniel Fekete. Melange: Space folding for multi-focus interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 1333–1342. ACM, 2008, doi:10.1145/1357054.1357263.
- [50] R. Etemadpour, P. Murray, and A. G. Forbes. Evaluating density-based motion for big data visual analytics. In *2014 IEEE International Conference on Big Data (Big Data)*, pages 451–460, Oct 2014, doi:10.1109/BigData.2014.7004262.
- [51] Gustav Fechner. *Elements of psychophysics. Vol. I*. American Psychological Association, 1966.
- [52] Chao Feng, Lyn Bartram, and Bernhard E. Riecke. Evaluating affective features of 3d motionscapes. In *Proceedings of the ACM Symposium on Applied Perception*, SAP '14, pages 23–30, New York, NY, USA, 2014. ACM, doi:10.1145/2628257.2628264.
- [53] Danyel Fisher. Animation for visualization: opportunities and drawbacks. In *Beautiful Visualization*, chapter 19, pages 329–352. O'Reilly Media, 2010.
- [54] Steven L. Franconeri and Daniel J. Simons. Moving and looming stimuli capture attention. *Perception & Psychophysics*, 65(7):999–1010, 2003, doi:10.3758/BF03194829.
- [55] Karl R. Gegenfurtner and Michael J. Hawken. Perceived velocity of luminance, chromatic and non-fourier stimuli: Influence of contrast and temporal frequency. *Vision Research*, 36(9):1281 – 1290, 1996, doi:10.1016/0042-6989(95)00198-0.

BIBLIOGRAPHY

- [56] Emilien Ghomi, Guillaume Faure, Stéphane Huot, Olivier Chapuis, and Michel Beaudouin-Lafon. Using rhythmic patterns as an input method. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1253–1262. ACM, 2012, doi:10.1145/2207676.2208579.
- [57] Gene Golovchinsky and Laurent Denoue. Moving markup: Repositioning freeform annotations. In *Proceedings of the Symposium on User Interface Software and Technology*, UIST '02, pages 21–30. ACM, 2002, doi:10.1145/571985.571989.
- [58] Gene Golovchinsky, Morgan N. Price, and Bill N. Schilit. From reading to retrieval: Freeform ink annotations as queries. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, pages 19–25. ACM, 1999, doi:10.1145/312624.312637.
- [59] Lars Grammel, Chris Bennett, Melanie Tory, and Margaret-Anne Storey. A Survey of Visualization Construction User Interfaces. In Mario Hlawitschka and Tino Weinkauf, editors, *EuroVis - Short Papers*, pages 19–23. The Eurographics Association, 2013, doi:10.2312/PE.EuroVisShort.EuroVisShort2013.019-023.
- [60] Lars Grammel, Melanie Tory, and Margaret-Anne Storey. How information visualization novices construct visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):943–952, 2010, doi:10.1109/TVCG.2010.164.
- [61] Thomas R. G. Green and Marian Petre. Usability analysis of visual programming environments: a 'cognitive dimensions' framework. *Journal of Visual Languages & Computing*, 7(2):131–174, 1996, doi:10.1006/jvlc.1996.0009.
- [62] John A. Greenwood and Mark Edwards. The detection of multiple global directions: Capacity limits with spatially segregated and transparent-motion signals. *Journal of Vision*, 9(1):40, 2009, doi:10.1167/9.1.40.
- [63] Steffen Hadlak, Heidrun Schumann, and Hans-Jörg Schulz. A Survey of Multifaceted Graph Visualization. In R. Borgo, F. Ganovelli, and I. Viola, editors, *Eurographics Conference on Visualization (EuroVis) - STARS*. The Eurographics Association, 2015, doi:10.2312/eurovisstar.20151109.
- [64] Gary Hardock, Gordon Kurtenbach, and William Buxton. A marking based interface for collaborative writing. In *Proceedings of the 6th Symposium on User Interface Software and Technology*, UIST '93, pages 259–266. ACM, 1993, doi:10.1145/168642.168669.
- [65] S. Haroz, K. L. Ma, and K. Heitmann. Multiple uncertainties in time-variant cosmological particle data. In *IEEE Pacific Visualization Symposium*, pages 207–214, March 2008, doi:10.1109/PACIFICVIS.2008.4475478.
- [66] Steve Haroz and David Whitney. Temporal thresholds for feature detection in flow visualization. In *Proceedings of the 7th Symposium on Applied Perception in Graphics and Visualization*, APGV '10, pages 163–163. ACM, 2010, doi:10.1145/1836248.1836285.

-
- [67] Steve Haroz and David Whitney. How capacity limits of attention influence information visualization effectiveness. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2402–2410, December 2012, doi:10.1109/TVCG.2012.233.
- [68] Björn Hartmann, Scott R. Klemmer, Michael Bernstein, Leith Abdulla, Brandon Burr, Avi Robinson-Mosher, and Jennifer Gee. Reflective physical prototyping through integrated design, test, and analysis. In *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology*, UIST '06, pages 299–308. ACM, 2006, doi:10.1145/1166253.1166300.
- [69] Jeffrey Heer and Danah Boyd. Vizster: Visualizing online social networks. In *Proceedings of the Proceedings of the IEEE Symposium on Information Visualization*, InfoVis '05, pages 32–39. IEEE, 2005, doi:10.1109/INFVIS.2005.39.
- [70] Fritz Heider and Marianne Simmel. An experimental study of apparent behavior. *The American Journal of Psychology*, 57(2):243–259, 1944, doi:10.2307/1416950.
- [71] Nathalie Henry, Jean-Daniel Fekete, and Michael McGuffin. Nodetrix: a hybrid visualization of social networks. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1302–1309, 2007, doi:10.1109/TVCG.2007.70582.
- [72] Nathalie Henry Riche, Tim Dwyer, Bongshin Lee, and Sheelagh Carpendale. Exploring the design space of interactive link curvature in network diagrams. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, pages 506–513, New York, NY, USA, 2012. ACM, doi:10.1145/2254556.2254652.
- [73] Ken Hinckley, Patrick Baudisch, Gonzalo Ramos, and Francois Guimbretiere. Design and analysis of delimiters for selection-action pen gesture phrases in scriboli. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, pages 451–460. ACM, 2005, doi:10.1145/1054972.1055035.
- [74] Ken Hinckley, Xiaojun Bi, Michel Pahud, and Bill Buxton. Informal information gathering techniques for active reading. In *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, pages 1893–1896. ACM, 2012, doi:10.1145/2207676.2208327.
- [75] Ken Hinckley, Francois Guimbretiere, Patrick Baudisch, Raman Sarin, Maneesh Agrawala, and Ed Cutrell. The springboard: Multiple modes in one spring-loaded control. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pages 181–190. ACM, 2006, doi:10.1145/1124772.1124801.
- [76] Ken Hinckley, Ken Hinckley, Shengdong Zhao, Raman Sarin, Patrick Baudisch, Edward Cutrell, Michael Shilman, and Desney Tan. Inkseine: In situ search for active note taking. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, pages 251–260. ACM, 2007, doi:10.1145/1240624.1240666.

BIBLIOGRAPHY

- [77] Ken Hinckley, Koji Yatani, Michel Pahud, Nicole Coddington, Jenny Rodenhouse, Andy Wilson, Hrvoje Benko, and Bill Buxton. Pen + touch = new tools. In *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology*, UIST '10, pages 27–36, New York, NY, USA, 2010. ACM, doi:10.1145/1866029.1866036.
- [78] Uta Hinrichs, Simon Butscher, Jens Müller, and Harald Reiterer. Diving in at the deep end: the value of alternative in-situ approaches for systematic library search. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 4634–4646. ACM, 2016, doi:10.1145/2858036.2858549.
- [79] J Hohnsbein and S Mateeff. The time it takes to detect changes in speed and direction of visual motion. *Vision Research*, 38(17):2569 – 2573, 1998, doi:http://dx.doi.org/10.1016/S0042-6989(98)00014-5.
- [80] Danny Holten, Petra Isenberg, Jarke J. van Wijk, and Jean-Daniel Fekete. An extended evaluation of the readability of tapered, animated, and textured directed-edge representations in node-link graphs. In *Proceedings of the 2011 IEEE Pacific Visualization Symposium*, PacificVis '11, pages 195–202. IEEE Computer Society, 2011, doi:10.1109/PACIFICVIS.2011.5742390.
- [81] Daniel E Huber and Christopher G Healey. Visualizing data with motion. In *Visualization, 2005. VIS 05. IEEE*, pages 527–534. IEEE, 2005, doi:10.1109/VISUAL.2005.1532838.
- [82] Samuel Huron, Sheelagh Cpendale, Alice Thudt, Anthony Tang, and Michael Mauerer. Constructive visualization. In *Proceedings of the Conference on Designing Interactive Systems*, DIS '14, pages 433–442. ACM, 2014, doi:10.1145/2598510.2598566.
- [83] Richard B Ivry and Asher Cohen. Asymmetry in visual search for targets defined by differences in movement speed. *Journal of Experimental Psychology: Human Perception and Performance*, 18:1045–1045, 1992, doi:10.1037/0096-1523.18.4.1045.
- [84] Paul Jaccard. The distribution of the flora in the alpine zone. *New phytologist*, 11(2):37–50, 1912, doi:10.1111/j.1469-8137.1912.tb05611.x.
- [85] Jaemin Jo, Sehi L'Yi, Bongshin Lee, and Jinwook Seo. TouchPivot: Blending wimp & post-wimp interfaces for data exploration on tablet devices. In *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, pages 2660–2671. ACM, 2017, doi:10.1145/3025453.3025752.
- [86] Ilir Jusufi, Yang Dingjie, and Andreas Kerren. The network lens: Interactive exploration of multivariate networks using visual filtering. In *International Conference Information Visualisation (IV)*, pages 35–42. IEEE, July 2010, doi:10.1109/IV.2010.15.
- [87] Hyunmo Kang, Catherine Plaisant, Bongshin Lee, and Benjamin B. Bederson. Netlens: Iterative exploration of content-actor network data. *Information Visualization*, 6(1):18–31, March 2007, doi:10.1057/palgrave.ivs.9500143.

-
- [88] Alison Kidd. The marks are on the knowledge worker. In *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, pages 186–191. ACM, 1994, doi:10.1145/191666.191740.
- [89] Nam Wook Kim, Nathalie Henry Riche, Benjamin Bach, Guanpeng Xu, Matthew Brehmer, Ken Hinckley, Michel Pahud, Haijun Xia, Michael J. McGuffin, and Hanspeter Pfister. Datatoon: Drawing dynamic network comics with pen + touch interaction. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, pages 105:1–105:12, New York, NY, USA, 2019. ACM, doi:10.1145/3290605.3300335.
- [90] Nam Wook Kim, Eston Schweickart, Zhicheng Liu, Mira Dontcheva, Wilmot Li, Jovan Popovic, and Hanspeter Pfister. Data-driven guides: Supporting expressive design for information graphics. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):491–500, January 2017, doi:10.1109/TVCG.2016.2598620.
- [91] David Kirsh. Thinking with external representations. *AI & Society*, 25(4):441–454, 2010, doi:10.1007/s00146-010-0272-8.
- [92] Bongshin Lee, Petra Isenberg, Nathalie Henry Riche, and Sheelagh Carpendale. Beyond mouse and keyboard: Expanding design considerations for information visualization interactions. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2689–2698, 2012, doi:10.1109/TVCG.2012.204.
- [93] Bongshin Lee, Greg Smith, Nathalie Henry Riche, Amy Karlson, and Sheelagh Carpendale. Sketchinsight: Natural data exploration on interactive whiteboards leveraging pen and touch interaction. In *Proc. of the IEEE Pacific Visualization Symposium*, pages 199–206. IEEE, 2015, doi:10.1109/PACIFICVIS.2015.7156378.
- [94] Jürg Lehni and Jonathan Puckey. Paper.js, 2018.
- [95] Guang Li, Xiang Cao, Sergio Paolantonio, and Feng Tian. Sketchcomm: a tool to support rich and flexible asynchronous communication of early design ideas. In *Proc. of the ACM Conf. on Computer Supported Cooperative Work*, pages 359–368. ACM, 2012, doi:10.1145/2145204.2145261.
- [96] Chunyuan Liao, Francois Guimbretiere, Ken Hinckley, Ken Hinckley, and Jim Hollan. Papiercraft: A gesture-based command system for interactive paper. *ACM Trans. Comput.-Hum. Interact.*, 14(4):18:1–18:27, January 2008, doi:10.1145/1314683.1314686.
- [97] Zhicheng Liu, John Thompson, Alan Wilson, Mira Dontcheva, James Delorey, Sam Grigg, Bernard Kerr, and John Stasko. Data illustrator: Augmenting vector design tools with lazy data binding for expressive visualization authoring. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 123:1–123:13. ACM, 2018, doi:10.1145/3173574.3173697.
- [98] M. Lockyer, L. Bartram, and B. E. Riecke. Simple motion textures for ambient affect. In *Proceedings of the International Symposium on Computational Aesthetics in Graphics, Visualization, and Imaging*, CAe '11, pages 89–96, New York, NY, USA, 2011. ACM, doi:10.1145/2030441.2030461.

BIBLIOGRAPHY

- [99] M Lockyer, L Bartram, T Schiphost, and K Studd. Enhancing visualization with expressive motion. *Electronic Imaging*, 2016(16):1–6, 2016, doi: 10.2352/ISSN.2470-1173.2016.16.HVEI-140.
- [100] David L MacAdam. Visual sensitivities to color differences in daylight. *Journal of the Optical Society of America*, 32(5):247–274, 1942, doi: 10.1364/JOSA.32.000247.
- [101] I. Scott MacKenzie and R. William Soukoreff. Phrase sets for evaluating text entry techniques. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '03, pages 754–755. ACM, 2003, doi: 10.1145/765891.765971.
- [102] Catherine C. Marshall. Annotation: From paper books to the digital library. In *Proceedings of the Second ACM International Conference on Digital Libraries*, DL '97, pages 131–140. ACM, 1997, doi: 10.1145/263690.263806.
- [103] Catherine C. Marshall. Toward an ecology of hypertext annotation. In *Proceedings of the Ninth ACM Conference on Hypertext and Hypermedia*, HYPERTEXT '98, pages 40–49. ACM, 1998, doi: 10.1145/276627.276632.
- [104] Catherine C. Marshall. *Reading and Writing the Electronic Book*. Morgan and Claypool Publishers, 2009.
- [105] S Mateeff, G Dimitrov, and J Hohnsbein. Temporal thresholds and reaction time to changes in velocity of visual motion. *Vision research*, 35(3):355–363, 1995, doi: 10.1016/0042-6989(94)00130-E.
- [106] Fabrice Matulic and Moira C. Norrie. Supporting active reading on pen and touch-operated tabletops. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, pages 612–619. ACM, 2012, doi: 10.1145/2254556.2254669.
- [107] M. J. McGuffin and I. Jurisica. Interaction techniques for selecting and manipulating subgraphs in network visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):937–944, Nov 2009, doi: 10.1109/TVCG.2009.151.
- [108] Peter McLeod, Jon Driver, and Jennie Crisp. Visual search for a conjunction of movement and form is parallel. *Nature*, 332(6160):154–155, 03 1988, doi: 10.1038/332154a0.
- [109] Tony McLoughlin, Robert S Laramée, Ronald Peikert, Frits H Post, and Min Chen. Over two decades of integration-based, geometric flow visualization. In *Computer Graphics Forum*, volume 29, pages 1807–1829. Wiley Online Library, 2010, doi: 10.1111/j.1467-8659.2010.01650.x.
- [110] Hrim Mehta, Adam James Bradley, Mark Hancock, and Christopher Collins. Metatation: Annotation as implicit interaction to bridge close and distant reading. *ACM Trans. on Computer-Human Interaction (TOCHI)*, pages 35:1–35:41, 2017, doi: 10.1145/3131609.

-
- [111] Gonzalo Gabriel Méndez, Uta Hinrichs, and Miguel A. Nacenta. Bottom-up vs. top-down: Trade-offs in efficiency, understanding, freedom and creativity with infovis tools. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, CHI '17, pages 841–852. ACM, 2017, doi:10.1145/3025453.3025942.
- [112] Gonzalo Gabriel Méndez, Miguel A. Nacenta, and Sebastien Vandenheste. Involver: Interactive visual language for visualization extraction and reconstruction. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 4073–4085. ACM, 2016, doi:10.1145/2858036.2858435.
- [113] A. Michotte. *The perception of causality*. Methuen's manuals of modern psychology. Basic Books, 1963.
- [114] Microsoft. Surface studio 2, 2018.
- [115] M. R. Morris, A. J. B. Brush, and B. R. Meyers. Reading revisited: Evaluating the usability of digital display surfaces for active reading tasks. In *IEEE International Workshop on Horizontal Interactive Human-Computer Systems (TABLETOP'07)*, pages 79–86, Oct 2007, doi:10.1109/TABLETOP.2007.12.
- [116] Pam A Mueller and Daniel M Oppenheimer. The pen is mightier than the keyboard: Advantages of longhand over laptop note taking. *Psychological Science*, 25(6):1159–1168, 2014, doi:10.1177/0956797614524581.
- [117] Till Nagel, Christopher Pietsch, and Marian Dörk. Staged analysis: From evocative to comparative visualizations of urban mobility. In *IEEE VIS Arts Program*, VISAP, pages 23–30, 2016.
- [118] Ken Nakayama and Gerald H. Silverman. Serial and parallel processing of visual feature conjunctions. *Nature*, 320(6059):264–265, 03 1986, doi:10.1038/320264a0.
- [119] C. Nobre, M. Meyer, M. Streit, and A. Lex. The state of the art in visualizing multivariate networks. *Computer Graphics Forum*, 38(3):807–832, 2019, doi:10.1111/cgf.13728.
- [120] Wouter De Nooy, Andrej Mrvar, and Vladimir Batagelj. *Exploratory Social Network Analysis with Pajek*. Cambridge University Press, 2011.
- [121] Kenton O'Hara and Abigail Sellen. A comparison of reading paper and on-line documents. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, CHI '97, pages 335–342. ACM, 1997, doi:10.1145/258549.258787.
- [122] Guy A. Orban, Frank Van Calenbergh, Bart De Bruyn, and Hugo Maes. Velocity discrimination in central and peripheral visual field. *Journal of the Optical Society of America A*, 2(11):1836–1847, Nov 1985, doi:10.1364/JOSAA.2.001836.

BIBLIOGRAPHY

- [123] Kurt Partridge, Saurav Chatterjee, Vibha Sazawal, Gaetano Borriello, and Roy Want. Tilttype: Accelerometer-supported text entry for very small devices. In *Proceedings of the 15th Annual ACM Symposium on User Interface Software and Technology*, UIST '02, pages 201–204, New York, NY, USA, 2002. ACM, doi:10.1145/571985.572013.
- [124] Ken Pfeuffer, Ken Hinckley, Michel Pahud, and Bill Buxton. Thumb + pen interaction on tablets. In *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, pages 3254–3266. ACM, 2017, doi:10.1145/3025453.3025567.
- [125] R. Pienta, F. Hohman, A. Endert, A. Tamersoy, K. Roundy, C. Gates, S. Navathe, and D. H. Chau. Vigor: Interactive visual exploration of graph query results. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):215–225, 2018, doi:10.1109/TVCG.2017.2744898.
- [126] Emmanuel Pietriga and Denis Barkats. Antenna dance at ALMA. <http://www.almaobservatory.org/en/announcement/antenna-dance-at-alma/> - Last accessed 2018-01-04, 2014.
- [127] Peter Pirolli and Stuart Card. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. pages 2–4, 2005.
- [128] Beryl Plimmer, Samuel Hsiao-Heng Chang, Meghavi Doshi, Laura Laycock, and Nilanthi Seneviratne. iannotate: Exploring multi-user ink annotation in web browsers. In *Proceedings of the Eleventh Australasian Conference on User Interface - Volume 106*, AUIIC '10, pages 52–60, Darlinghurst, Australia, Australia, 2010. Australian Computer Society, Inc.
- [129] Sriram Ramachandran and Ramanujan Kashi. An architecture for ink annotations on web documents. In *Proceedings of the Int. Conference on Document Analysis and Recognition*, ICDAR '03, pages 256–260. IEEE, 2003, doi:10.1109/ICDAR.2003.1227669.
- [130] Jef Raskin. *The Humane Interface: New Directions for Designing Interactive Systems*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 2000.
- [131] Donghao Ren, Tobias Höllerer, and Xiaoru Yuan. ivisdesigner: Expressive interactive design of information visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2092–2101, December 2014, doi:10.1109/TVCG.2014.2346291.
- [132] Donghao Ren, Bongshin Lee, and Matthew Brehmer. Charticulator: Interactive construction of bespoke chart layouts. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):789–799, Jan 2019, doi:10.1109/TVCG.2018.2865158.
- [133] Nathalie Henry Riche, Tim Dwyer, Bongshin Lee, and Sheelagh Carpendale. Exploring the design space of interactive link curvature in network diagrams. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, pages 506–513. ACM, 2012, doi:10.1145/2254556.2254652.

-
- [134] Yann Riche, Nathalie Henry Riche, Ken Hinckley, Sheri Panabaker, Sarah Fuelling, and Sarah Williams. As we may ink?: Learning from everyday analog pen use to improve digital ink experiences. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, CHI '17, pages 3241–3253. ACM, 2017, doi:10.1145/3025453.3025716.
- [135] Hugo Romat, Caroline Appert, Benjamin Bach, Nathalie Henry-Riche, and Emmanuel Pietriga. Animated edge textures in node-link diagrams: A design space and initial evaluation. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 187:1–187:13. ACM, 2018, doi:10.1145/3173574.3173761.
- [136] Hugo Romat, Nathalie Henry Riche, Ken Hinckley, Bongshin Lee, Caroline Appert, Emmanuel Pietriga, and Christopher Collins. Activeink: (th)inking with data. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, CHI '19, pages 42:1–42:13. ACM, 2019, doi:10.1145/3290605.3300272.
- [137] Hugo Romat, Dylan Lebout, Emmanuel Pietriga, and Caroline Appert. Animated particles in node-link diagrams: Influence of particles' visual appearance on their perceived speed. In *INTERACT'19: IFIP International Conference on Human-Computer Interaction*, Paphos, Cyprus, September 2019. Springer.
- [138] Hugo Romat, Emmanuel Pietriga, Nathalie Henry Riche, Ken Hinckley, and Caroline Appert. Spaceink: Making space for in-context annotations. In *Proceedings of the 32th Annual ACM Symposium on User Interface Software and Technology*, UIST '19, pages 107–112, New York, NY, USA, 2019. ACM.
- [139] Volker Roth and Thea Turner. Bezel swipe: Conflict-free scrolling and multiple selection on mobile touch screen devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages 1523–1526. ACM, 2009, doi:10.1145/1518701.1518933.
- [140] Quentin Roy, Futian Zhang, and Daniel Vogel. Automation Accuracy Is Good, but High Controllability May Be Better. In *to appear in CHI '19: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Glasgow, UK, 2019. ACM Press, doi:10.1145/3290605.3300750.
- [141] Vít Rusnák, Caroline Appert, Olivier Chapuis, and Emmanuel Pietriga. Designing coherent gesture sets for multi-scale navigation on tablets. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 142:1–142:12. ACM, 2018, doi:10.1145/3173574.3173716.
- [142] Jeffrey M. Rzeszotarski and Aniket Kittur. Kinetica: Naturalistic multi-touch data visualization. In *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, pages 897–906. ACM, 2014, doi:10.1145/2556288.2557231.
- [143] Ramik Sadana and John Stasko. Designing and implementing an interactive scatterplot visualization for a tablet computer. In *Proc. of the Int. Working Conf. on Advanced Visual Interfaces*, pages 265–272. ACM, 2014, doi:10.1145/2598153.2598163.

BIBLIOGRAPHY

- [144] Ramik Sadana and John Stasko. Designing multiple coordinated visualizations for tablets. *Computer Graphics Forum*, 35(3):261–270, June 2016, doi:10.1111/cgf.12902.
- [145] Arvind Satyanarayan and Jeffrey Heer. Lyra: An interactive visualization design environment. *Computer Graphics Forum*, 33(3):351–360, 2014, doi:10.1111/cgf.12391.
- [146] Arvind Satyanarayan, Dominik Moritz, Kanit Wongsuphasawat, and Jeffrey Heer. Vega-lite: A grammar of interactive graphics. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):341–350, 2017.
- [147] Eric Saund and Edward Lank. Stylus input and editing without prior selection of mode. In *Proceedings of the Symposium on User Interface Software and Technology*, UIST '03, pages 213–216. ACM, 2003, doi:10.1145/964696.964720.
- [148] Michael Scaife and Yvonne Rogers. External cognition: How do graphical representations work? *Int. J. Hum.-Comput. Stud.*, 45:185–213, August 1996, doi:10.1006/ijhc.1996.0048.
- [149] Joey Scarr, Andy Cockburn, Carl Gutwin, and Andrea Bunt. Improving command selection with CommandMaps. In *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, pages 257–266. ACM, 2012, doi:10.1145/2207676.2207713.
- [150] Roeland Scheepens, Christophe Hurter, Huub Van De Wetering, and Jarke J Van Wijk. Visualization, selection, and analysis of traffic flows. *IEEE transactions on visualization and computer graphics*, 22(1):379–388, 2016, doi:10.1109/TVCG.2015.2467112.
- [151] Bill N. Schilit, Gene Golovchinsky, and Morgan N. Price. Beyond paper: Supporting active reading with free form digital ink annotations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '98, pages 249–256. ACM Press/Addison-Wesley Publishing Co., 1998, doi:10.1145/274644.274680.
- [152] Jason M. Scimeca and Steven L. Franconeri. Selecting and tracking multiple objects. *Wiley Interdisciplinary Reviews: Cognitive Science*, 6(2):109–118, 2015, doi:10.1002/wcs.1328.
- [153] R. Sekuler, S.N.J. Watamaniuk, and R. Blake. Perception of visual motion. In *Stevens' Handbook of Experimental Psychology: Vol. 1 Sensation and perception (3rd edition)*, page 121–176. Wiley & Sons, 2004.
- [154] Abigail J. Sellen and Richard H.R. Harper. *The Myth of the Paperless Office*. MIT Press, Cambridge, MA, USA, 2003.
- [155] Microsoft Cognitive Services. Ink analysis api, 2018.
- [156] Zeqian Shen, Kwan-Liu Ma, and Tina Eliassi-Rad. Visual analysis of large heterogeneous social networks by semantic and structural abstraction. *IEEE Transactions on Visualization and Computer Graphics*, 12(6):1427–1439, 2006, doi:10.1109/TVCG.2006.107.

-
- [157] Lei Shi, Qi Liao, Hanghang Tong, Yifan Hu, Yue Zhao, and Chuang Lin. Hierarchical focus+context heterogeneous network visualization. In *Pacific Visualization Symposium*, pages 89–96. IEEE, March 2014, doi:10.1109/PacificVis.2014.44.
- [158] Hirohito Shibata, Kentaro Takano, and Kengo Omura. Comparison of paper and computer displays in reading including frequent movement between pages. In *Proceedings of the Australian Computer-Human Interaction Conference, OzCHI '14*, pages 549–558. ACM, 2014, doi:10.1145/2686612.2686700.
- [159] B. Shneiderman. Direct manipulation: A step beyond programming languages. *IEEE Computer*, 16(8):57–69, August 1983, doi:10.1109/MC.1983.1654471.
- [160] Ben Shneiderman. Dynamic queries for visual information seeking. *IEEE Softw.*, 11(6):70–77, November 1994, doi:10.1109/52.329404.
- [161] Ben Shneiderman and Aleks Aris. Network visualization by semantic substrates. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):733–740, 2006, doi:10.1109/TVCG.2006.166.
- [162] Ben Shneiderman and Cody Dunne. Interactive network exploration to derive insights: Filtering, clustering, grouping, and simplification. In *Proceedings of the 20th International Conference on Graph Drawing, GD'12*, pages 2–18. Springer-Verlag, 2013, doi:10.1007/978-3-642-36763-2_2.
- [163] David R. R. Smith and Andrew M. Derrington. What is the denominator for contrast normalisation? *Vision Research*, 36(23):3759 – 3766, 1996, doi:10.1016/0042-6989(96)00100-9.
- [164] Marc A. Smith, Ben Shneiderman, Natasa Milic-Frayling, Eduarda Mendes Rodrigues, Vladimir Barash, Cody Dunne, Tony Capone, Adam Perer, and Eric Gleave. Analyzing (Social Media) Networks with NodeXL. In *Proceedings of the 4th International Conference on Communities and Technologies*, pages 255–264. ACM, 2009, doi:10.1145/1556460.1556497.
- [165] Andre Spritzer, Jeremy Boy, Pierre Dragicevic, Jean-Daniel Fekete, and Carla Maria Dal Sasso Freitas. Towards a smooth design process for static communicative node-link diagrams. *Computer Graphics Forum*, 34(3):461–470, 2015, doi:10.1111/cgf.12658.
- [166] D. A. Szafir. Modeling color difference for visualization design. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):392–401, Jan 2018, doi:10.1109/TVCG.2017.2744359.
- [167] Tableau. Tableau Desktop. <https://www.tableau.com>, 2019. [online; accessed 2019-03-15].
- [168] Craig Tashman, Cristiano Ghersi, Stephen Dukker, and Dalas Verdugo. Liquidtext. <https://www.liquidtext.net/>, 2010. Accessed: 2019-03-21.
- [169] Craig S. Tashman and W. Keith Edwards. Liquidtext: A flexible, multitouch environment to support active reading. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '11*, pages 3285–3294. ACM, 2011, doi:10.1145/1978942.1979430.

BIBLIOGRAPHY

- [170] Tatiana Tekušová, Jörn Kohlhammer, Slawomir J. Skwarek, and Galina V. Paramei. Perception of direction changes in animated data visualization. In *Proceedings of the 5th Symposium on Applied Perception in Graphics and Visualization*, APGV '08, pages 205–205. ACM, 2008, doi:10.1145/1394281.1394331.
- [171] Kim Thomas. Just noticeable difference and tempo change. *Journal of Scientific Psychology*, 2:14–20, 2007.
- [172] Stefan Tilkov and Steve Vinoski. Node.js: Using javascript to build high-performance network programs. *IEEE Internet Computing*, 14(6):80–83, 2010, doi:10.1109/MIC.2010.145.
- [173] Christian Tominski, Stefan Gladisch, Ulrike Kister, Raimund Dachselt, and Heidrun Schumann. A Survey on Interactive Lenses in Visualization. In R. Borgo, R. Maciejewski, and I. Viola, editors, *EuroVis - STARs*. The Eurographics Association, 2014, doi:10.2312/eurovisstar.20141172.
- [174] Anne Treisman. Preattentive processing in vision. *Comput. Vision Graph. Image Process.*, 31(2):156–177, August 1985, doi:10.1016/S0734-189X(85)80004-9.
- [175] Barbara Tversky, Julie Bauer Morrison, and Mireille Betrancourt. Animation: Can it facilitate? *Int. J. Hum.-Comput. Stud.*, 57(4):247–262, October 2002, doi:10.1006/ijhc.2002.1017.
- [176] Stef van den Elzen and Jarke van Wijk. Multivariate network exploration and presentation: From detail to overview via selections and aggregations. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2310–2319, 2014, doi:10.1109/TVCG.2014.2346441.
- [177] Frank van Ham and Adam Perer. “search, show context, expand on demand”: Supporting large graph exploration with degree-of-interest. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):953–960, 2009, doi:10.1109/TVCG.2009.108.
- [178] Fernanda Viegas, Martin Wattenberg, Frank van Ham, Jesse Kriss, and Matt McKeon. Manyeyes: a site for visualization at internet scale. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1121–1128, 2007, doi:10.1109/TVCG.2007.70577.
- [179] 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. *Computer Graphics Forum*, 30(6):1719–1749, 2011, doi:10.1111/j.1467-8659.2011.01898.x.
- [180] Jagoda Walny, Samuel Huron, Charles Perin, Tiffany Wun, Richard Pusch, and Sheelagh Cpendale. Active reading of visualizations. *IEEE Trans. on Visualization and Computer Graphics*, 24(1):770–780, 2018, doi:10.1109/TVCG.2017.2745958.
- [181] Colin Ware and Robert Bobrow. Motion to support rapid interactive queries on node-link diagrams. *ACM Trans. Appl. Percept.*, 1(1):3–18, July 2004, doi:10.1145/1008722.1008724.

- [182] Colin Ware, Joseph Bonner, William Knight, and Rod Cater. Moving icons as a human interrupt. *International Journal of Human-Computer Interaction*, 4(4):341–348, 1992, doi:10.1080/10447319209526047.
- [183] Colin Ware, Helen Purchase, Linda Colpoys, and Matthew McGill. Cognitive measurements of graph aesthetics. *Information Visualization*, 1(2):103–110, 2002, doi:10.1057/palgrave.ivs.9500013.
- [184] Martin Wattenberg. Visual exploration of multivariate graphs. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pages 811–819. ACM, 2006, doi:10.1145/1124772.1124891.
- [185] D. Weiskopf. On the role of color in the perception of motion in animated visualizations. In *IEEE Visualization 2004*, pages 305–312, Oct 2004, doi:10.1109/VISUAL.2004.73.
- [186] Daniel Wigdor and Ravin Balakrishnan. Tilttext: Using tilt for text input to mobile phones. In *Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology*, UIST '03, pages 81–90, New York, NY, USA, 2003. ACM, doi:10.1145/964696.964705.
- [187] Alexandra Willis and Stephen J Anderson. Colour and luminance interactions in the visual perception of motion. *Proceedings of the Royal Society of London B: Biological Sciences*, 269(1495):1011–1016, 2002, doi:10.1098/rspb.2002.1985.
- [188] Haijun Xia, Bruno Araujo, Tovi Grossman, and Daniel Wigdor. Object-oriented drawing. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 4610–4621, New York, NY, USA, 2016. ACM, doi:10.1145/2858036.2858075.
- [189] Haijun Xia, Bruno Araujo, and Daniel Wigdor. Collection objects: Enabling fluid formation and manipulation of aggregate selections. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, pages 5592–5604, New York, NY, USA, 2017. ACM, doi:10.1145/3025453.3025554.
- [190] Haijun Xia, Nathalie Henry Riche, Fanny Chevalier, Bruno De Araujo, and Daniel Wigdor. Dataink: Direct and creative data-oriented drawing. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 223:1–223:13. ACM, 2018, doi:10.1145/3173574.3173797.
- [191] Haijun Xia, Ken Hinckley, Michel Pahud, Xiao Tu, and Bill Buxton. Writlarge: Ink unleashed by unified scope, action, & zoom. In *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, pages 3227–3240. ACM, 2017, doi:10.1145/3025453.3025664.
- [192] Zixin Yong and Po-Jang Hsieh. Speed–size illusion correlates with retinal-level motion statistics. *Journal of vision*, 17(9):1–1, 2017.
- [193] Dongwook Yoon, Nicholas Chen, and Francois Guimbretiere. Texttearing: Opening white space for digital ink annotation. In *Proceedings of the Symposium on User Interface Software and Technology*, UIST '13, pages 107–112. ACM, 2013, doi:10.1145/2501988.2502036.

BIBLIOGRAPHY

- [194] Dongwook Yoon, Nicholas Chen, Francois Guimbretiere, and Abigail Sellen. Richreview: Blending ink, speech, and gesture to support collaborative document review. In *Proceedings of the Symposium on User Interface Software and Technology*, UIST '14, pages 481–490. ACM, 2014, doi:10.1145/2642918.2647390.
- [195] Robert Zeleznik and Timothy Miller. Fluid inking: Augmenting the medium of free-form inking with gestures. In *Proceedings of Graphics Interface 2006*, GI '06, pages 155–162. Canadian Information Processing Society, 2006.
- [196] Emanuel Zgraggen, Robert Zeleznik, and Steven M Drucker. PanoramicData: Data analysis through pen & touch. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2112–2121, 2014, doi:10.1109/TVCG.2014.2346293.
- [197] Shengdong Zhao, Maneesh Agrawala, and Ken Hinckley. Zone and polygon menus: Using relative position to increase the breadth of multi-stroke marking menus. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pages 1077–1086, New York, NY, USA, 2006. ACM, doi:10.1145/1124772.1124933.
- [198] Zhenpeng Zhao, Sriram Karthik Badam, Senthil Chandrasegaran, Deok Gun Park, Niklas L.E. Elmqvist, Lorraine Kisselburgh, and Karthik Ramani. skwiki: A multimedia sketching system for collaborative creativity. In *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, pages 1235–1244. ACM, 2014, doi:10.1145/2556288.2557394.

APPENDIX **A**



Appendix

A.1 Detailed instructions participants received during Graphies' study

Instructions

Welcome and thank you for participating in this study.
All results gathered during this test will be anonymized.

Feel free to make comments aloud, and to ask questions to the experimenter.

Experiment Description

This experiment aims at evaluating whether *Graphies* enables the design of engaging and expressive node-link diagram visualizations.

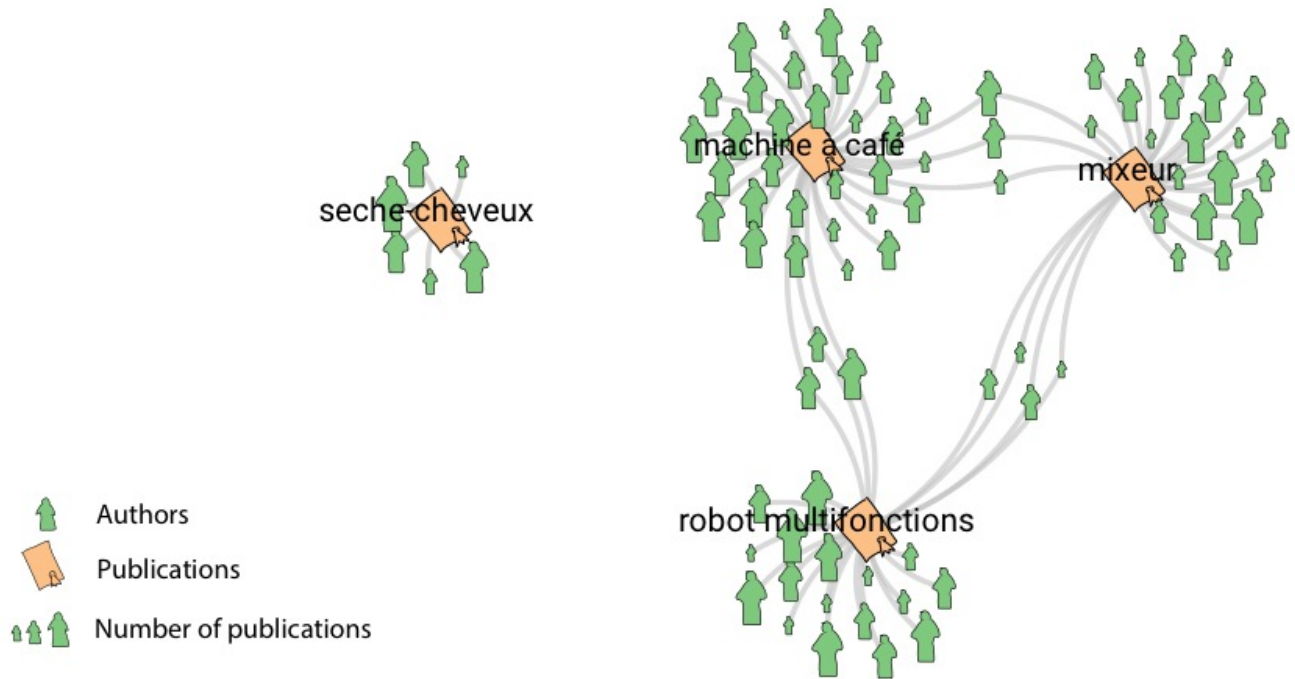
The experiment is organized in three steps:

- 1- You will watch a short demonstration perform by the instructor and you will have to reproduce some specific actions to personalize the graph visualization.
- 2- Then comes the main experiment. It consists of a series of 3 tasks. Each task is similar to the training task: you will be presented with a target visualization, that you will have to replicate. For each visualization, a short text explains what data the visualization shows. After each task, you will complete a short questionnaire.
- 3- Finally, we will ask you to perform a more open-ended task, in which you will be free to design the visualization of your choice for the "InfoVis" dataset. Design the visualization that you consider good to:
 - a. show the most productive actors (i.e., authors who have a significant number of articles)
 - b. show their articles, and visually emphasize the ones that have a high number of citations
 - c. [optional] illustrate the trend "when important actors collaborate, the resulting articles have a high impact"

At the end of the experiment, you will fill in another questionnaire.

Thank you!

Use Case 1



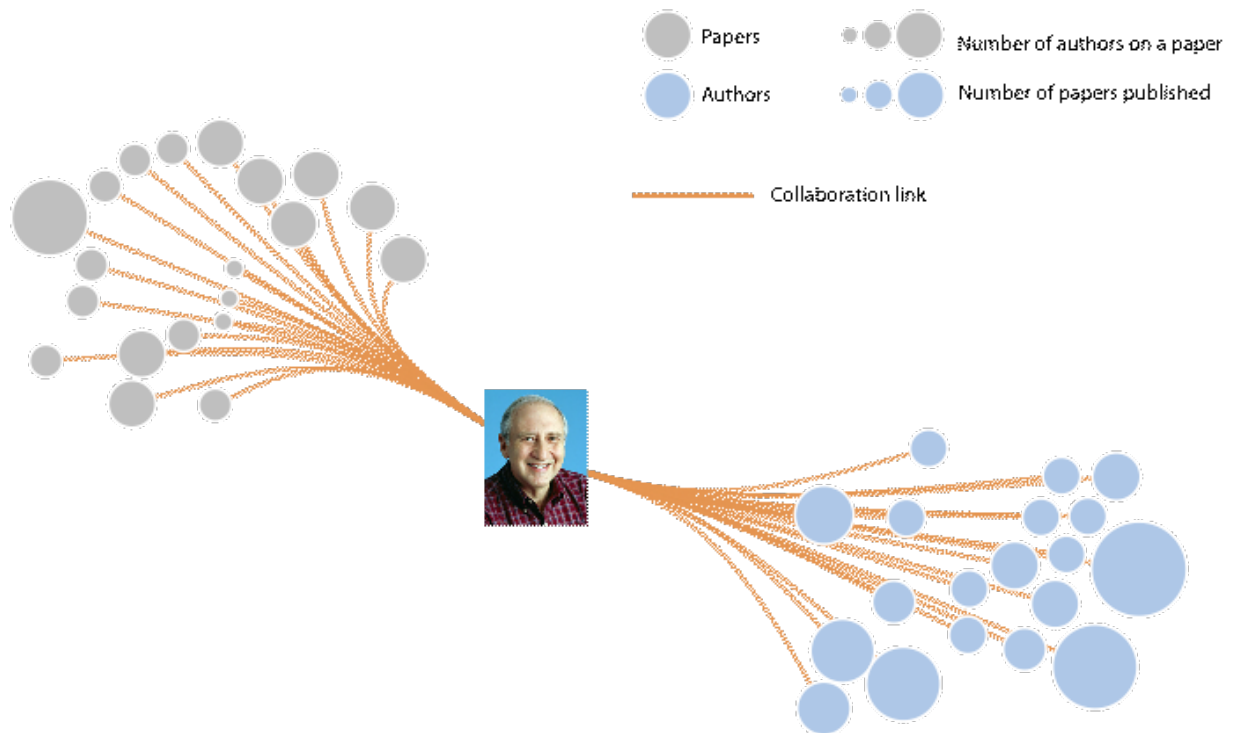
In this use case, you will be asked to:

- show authors of documents about home appliances;
- emphasize the most important authors (based on the number of publications they have authored).

Steps:

- 1) First, show the four documents about “machine a cafe”, “seche cheveux”, “mixeur” and “robot multifonctions”.
- 2) Then, show authors who have collaborated on those documents.
- 3) Represent authors and documents with glyphs resembling the ones shown in the above figure. Make the size of authors dependent on their number of publications, and the labels of the document nodes visible.
- 4) When satisfied with the result, save your visualization.

Use case 2



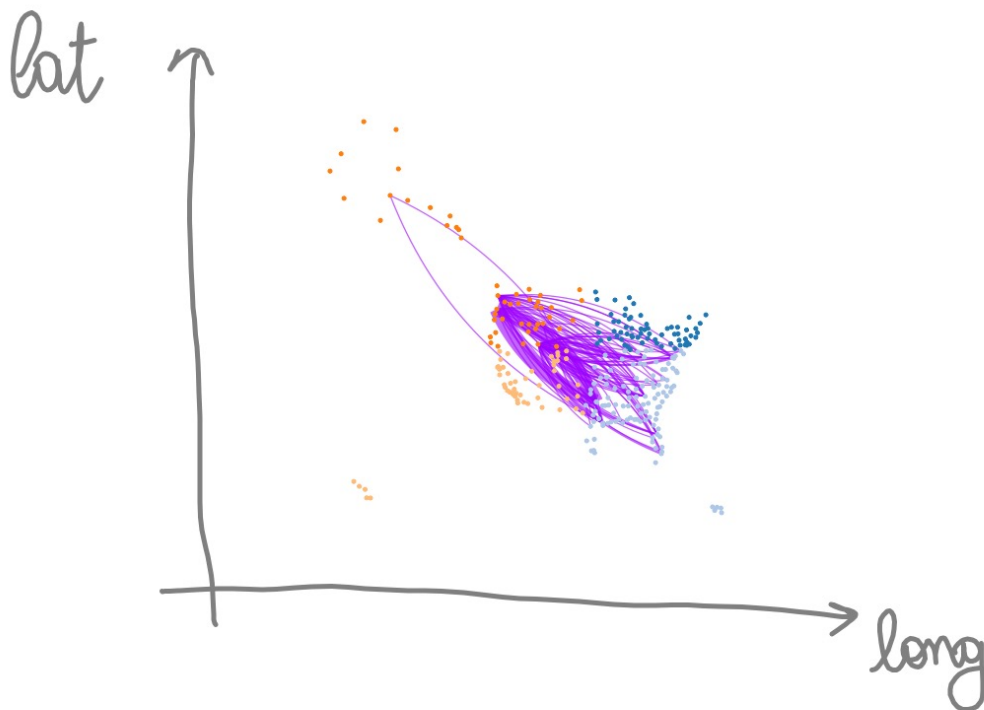
In this use case, you will be asked to represent all of Ben Shneiderman's papers, as well as all his co-authors.

Steps:

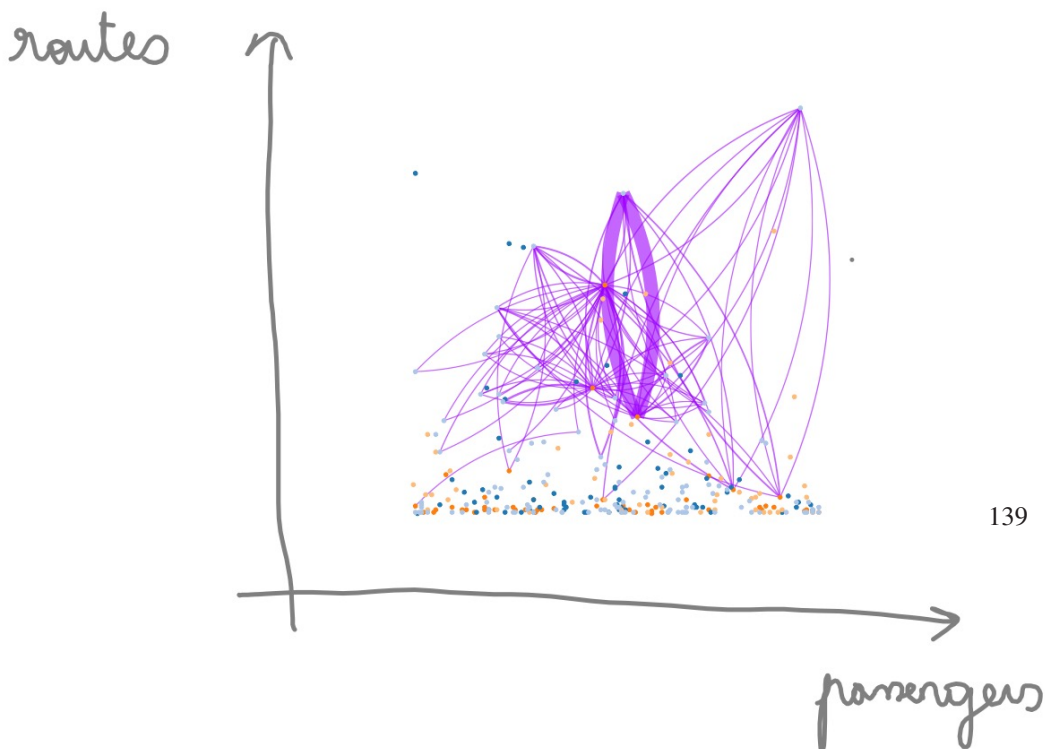
- 1) Show Ben Shneiderman with the picture that we make available on the file system.
- 2) Show all his co-authors and all his papers.
- 3) Make the size of nodes representing papers depend on their number of authors.
- 4) Make the size of nodes representing authors depend on their numbers of papers.
- 5) Finally, set the links' color to orange, and bundle edges to get a result similar to the one in the above figure.
- 6) When satisfied with the result, save your visualization.

Use case 3

In this use case, you will be asked to replicate the two visualizations (A) and (B) above, as well as a video transitioning between the two.



A



B

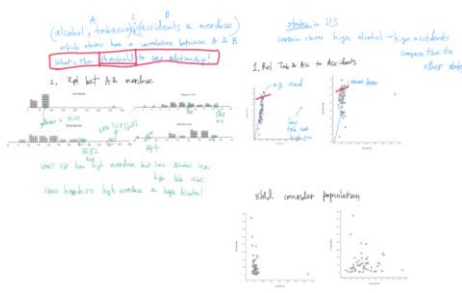
Steps:

- For Visualization (A):
 1. Show all airports.
 2. Record a frame in the video editor ([video frame 1](#))
 3. Lay out nodes according to their geographical location (longitude on the x-axis, latitude on the y-axis).
 4. Show routes (connections) between the South East and the North West airports only.
 5. Make links purple, and set their stroke size to depend on the number of carriers.
 6. Sketch x- and y- axes, and their labels.
 7. When satisfied with the result, record it as frame in the video editor ([video frame 2](#)).

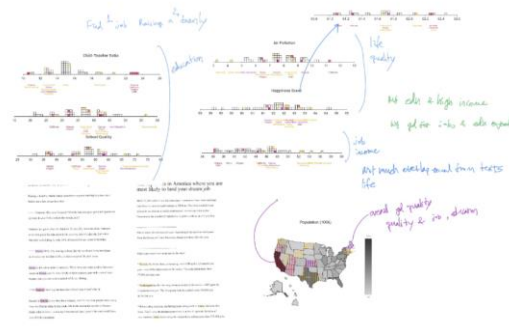
- For Visualization (B):
 1. Change the nodes' layout: make their location on the x-axis depend on the number of passengers, and their location on the y-axis depend on the number of routes (liaisons) going through them.
 2. Sketch the new x- and y-axes, and their labels.
 3. When satisfied with the result, record it as frame in the video editor ([video frame 3](#)).
 4. Export the 3-frame video.

A.2. Canvas created by participants during ActiveInk study

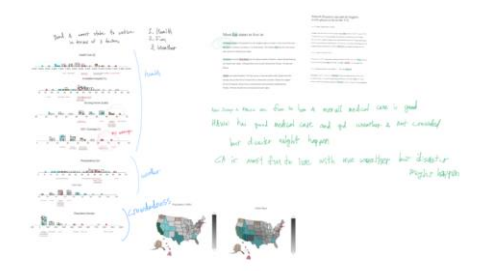
A.2 Canvas created by participants during ActiveInk study



P1- Inking



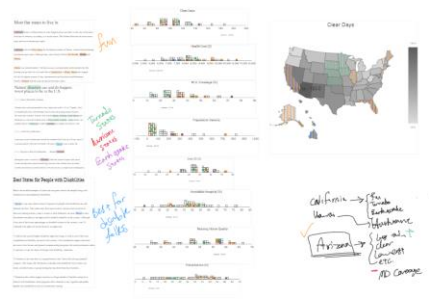
P1- Prefix



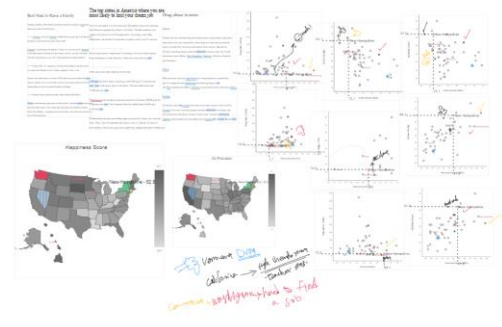
P1- Postfix



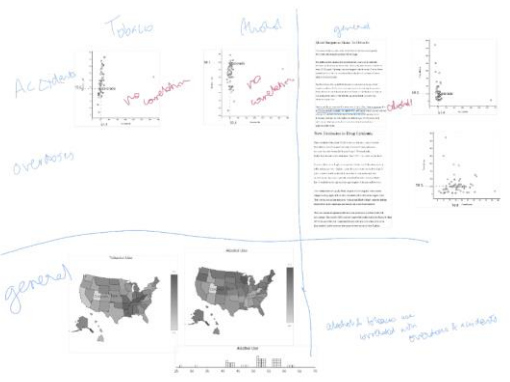
P2- Inking



P2- Prefix



P2- Postfix



P3- Inking

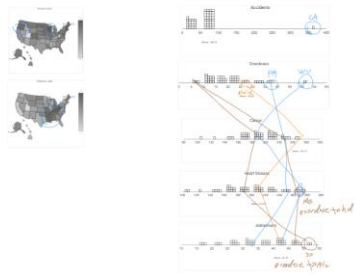


P3- Prefix

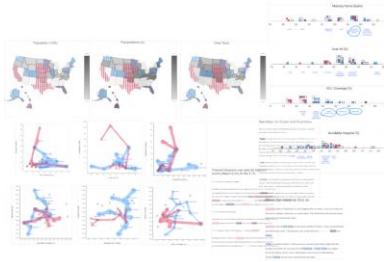


P3- Postfix

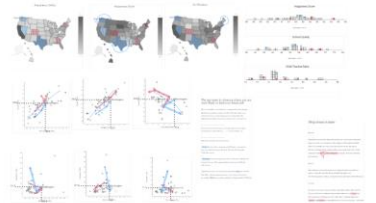
A. APPENDIX



P4- Inking



P4- Prefix



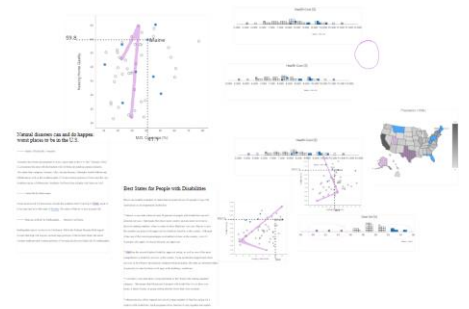
P4- Postfix



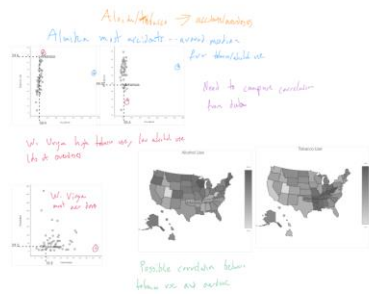
P5- Inking



P5- Prefix



P5- Postfix



P6- Inking

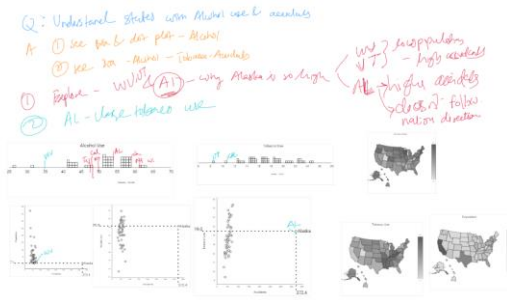


P6- Prefix



P6- Postfix

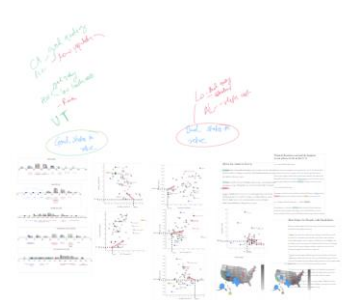
A.2. Canvas created by participants during ActiveInk study



P7- Inking



P7- Prefix



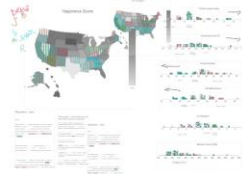
P7- Postfix



P8- Inking



P8- Prefix



P8- Prefix



P8- Postfix



P8- Postfix

Titre : De l'exploration des données à la présentation : concevoir de nouveaux systèmes et techniques d'interaction pour améliorer la création de sens à partir de données

Mots clés : Stylet+Gestes, Manipulation de données, Techniques d'interaction

Résumé :

Au cours de la dernière décennie, la quantité de données n'a cessé d'augmenter. Ces données peuvent provenir de sources variées, telles que des smartphones, des enregistreurs audio, des caméras, des capteurs, des simulations, et peuvent avoir différentes structures. Bien que les ordinateurs puissent nous aider à traiter ces données, c'est le jugement et l'expertise humaine qui les transforment réellement en connaissances. Cependant, pour donner un sens à ces données de plus en plus diversifiées, des techniques de visualisation et d'interaction sont nécessaires. Ce travail de thèse contribue de telles techniques pour faciliter l'exploration et la présentation des données, lors d'activités visant à faire sens des données.

Dans la première partie de cette thèse, nous nous concentrons sur les systèmes interactifs et les techniques d'interaction pour aider les utilisateurs à faire sens des données. Nous étudions comment les utilisateurs travaillent avec des contenus divers afin de leur permettre d'externaliser leurs pensées par le biais d'annotations digitales. Nous présentons notre approche avec deux systèmes. Le premier, ActiveInk, permet l'utilisation naturelle du stylet pour la

lecture active, lors d'un processus d'exploration de données. Le second système, Spacelnk, est un espace de conception de techniques en utilisant le stylet et les gestes, qui permet de créer de l'espace pour les annotations, pendant la lecture active, en ajustant dynamiquement le contenu du document.

Dans la deuxième partie de cette thèse, nous avons étudié les techniques permettant de représenter visuellement les éléments de réponses aux questions quand les utilisateurs essaient de faire sens des données. Nous nous concentrons sur l'une des structures de données les plus élaborées : les réseaux multi-variés, que nous visualisons à l'aide de diagrammes noeuds-liens. Nous présentons d'abord un système, Graphies, qui permet la création de visualisations expressives de diagrammes noeuds-liens en fournissant aux concepteurs un environnement de travail flexible qui rationalise le processus créatif et offre un support efficace pour les itérations rapides de conception. Allant au-delà de l'utilisation de variables visuelles statiques dans les diagrammes noeuds-liens, nous avons étudié le potentiel des variables liées au mouvement pour encoder les attributs des données.

Title : From data exploration to presentation: designing new systems and interaction techniques to enhance the sense-making process

Keywords : Pen+Touch, Data manipulation, Interaction Techniques

Abstract : During the last decade, the amount of data has been constantly increasing. These data can come from several sources such as smartphones, audio recorders, cameras, sensors, simulations, and can have various structure. While computers can help us process these data, human judgment and domain expertise is what turns the data into actual knowledge. However, making sense of this increasing amount of diverse data requires visualization and interaction techniques. This thesis contributes such techniques to facilitate data exploration and presentation, during sense-making activities.

In the first part of this thesis, we focus on interactive systems and interaction techniques to support sense-making activities. We investigate how users work with diverse content in order to make them able to externalize thoughts through digital annotations. We present our approach with two systems. The first system, ActiveInk enables the natural use of pen for active rea-

ding during a data exploration process. The second system, Spacelnk, is a design space of pen & touch techniques that make space for in-context annotations during active reading by dynamically reflowing documents.

In the second part, we focus on techniques to visually represent insights and answers to questions that arise during sense-making activities. We focus on one of the most elaborate data structures: multivariate networks, that we visualize using a node-link diagram visualization. We first present a system, Graphies, that enables the creation of expressive node-link diagram visualizations by providing designers with a flexible workflow that streamlines the creative process, and effectively supports quick design iterations. Moving beyond the use of static visual variables in node-link diagrams, we investigated the use of motion to encode data attributes.

