



Languages and Interaction Techniques for the Visualization and Manipulation of Massive Datasets

Emmanuel Pietriga

► To cite this version:

Emmanuel Pietriga. Languages and Interaction Techniques for the Visualization and Manipulation of Massive Datasets. Human-Computer Interaction [cs.HC]. Université Paris Sud - Paris XI, 2012. tel-00709533

HAL Id: tel-00709533

<https://theses.hal.science/tel-00709533>

Submitted on 18 Jun 2012

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.

UNIVERSITÉ PARIS-SUD

HABILITATION À DIRIGER DES RECHERCHES

présentée par

Emmanuel Pietriga

Spécialité : Informatique – Interaction Homme-Machine

Languages and Interaction Techniques for the Visualization and Manipulation of Massive Datasets

8 juin 2012

Lynda Hardman	CWI	Examineur
Robert J.K. Jacob	Tufts University	Rapporteur
David R. Karger	MIT	Examineur
Wendy E. Mackay	INRIA	Examineur
Laurence Nigay	Université Joseph Fourier	Rapporteur
Christine Paulin	Université Paris-Sud	Examineur
Claude Puech	Université Paris-Sud & INRIA	Examineur
Mary Beth Rosson	Pennsylvania State University	Rapporteur

Habilitation à Diriger des Recherches préparée au sein de l'INRIA Saclay – Île-de-France
et du Laboratoire de Recherche en Informatique de l'Université Paris-Sud

Acknowledgments

I would first like to thank the members of my Habilitation committee. The three *rappor-teurs*: Rob Jacob, Laurence Nigay and Mary Beth Rosson; and the five *examineurs*: Lynda Hardman, David Karger, Wendy Mackay, Christine Paulin and Claude Puech. I would also like to thank Ravin Balakrishnan, Pierre-Henri Cubaud and Claude Puech for supporting my application to the Habilitation à Diriger des Recherches.

Thanks to Michel Beaudouin-Lafon and Wendy Mackay for having given me the opportunity to join the In-Situ project-team. Special thanks again to Claude Puech, who has played a pivotal role in the recent projects I have initiated, related to the collaboration with ALMA and the Massive Data project at INRIA Chile.

I would also like to thank all members of In-Situ. First and foremost Caroline Appert and Olivier Chapuis, who have been excellent colleagues and friends for more than eight years; Romain Primet who has been collaborating with me on several projects; the PhD students I have been co-advising: Benjamin Bach, Cyprien Pindat, Mathieu Nancel, and the other students whom I have had great pleasure working with: Olivier Bau, Jean-Baptiste Labrune, Julie Wagner, Jian Zhao; the other past-and-present team members that I have worked with throughout the years: Rodrigo de Almeida, Anastasia Bezerianos, Stéphane Huot, Julien Husson, Ilaria Lliccardi, The Nhan Luong, Clément Pillias, Daniel Spelmezan, Theophanis Tsandilas; many Aviz team members, including: Jean-Daniel Fekete, Fanny Chevalier (merci), Nathalie Henry, Tomer Moscovich; and colleagues beyond INRIA: Ravin Balakrishnan, Chris Bizer, Andy Cockburn, Sarah Cohen-Boulakia, Christine Froidevaux, David Karger, Ryan Lee, René Vestergaard, with a special mention to ALMA colleagues: Joseph Schwarz, Marcus Schilling, Denis Barkats, Emilio Barrios, Baltasar Vila-Vilaro.

Many thanks to Alexandra Merlin who has been helping me a lot throughout the last two years and made my life as In-Situ team leader very easy.

Finally, I would like to thank my family for their patience, and Maxime & Thomas for being such great kids.

Table of Contents

1	Introduction	5
1.1	Visualization and Transformation of Semistructured Data	9
1.2	Multi-scale Navigation in Large Datasets	10
2	Techniques for the Visualization of Semistructured Data	13
2.1	Graphical Transformation and Representation of Semantic Web Data	16
2.1.1	Multi-scale Visualization of RDF Graphs	16
2.1.2	Graph Style Sheets	18
2.1.3	Visualization of Populated Ontologies	22
2.2	A Presentation Vocabulary for the Web of Data	24
2.3	Graph Visualization	28
2.3.1	Multi-scale Visualization of Large Graphs	28
2.3.2	Interactive Navigation in Large Graphs	29
3	Multi-scale Navigation in Large Datasets	33
3.1	Design and Evaluation of Interaction Techniques	35
3.1.1	An Operationalization of Multi-scale Search	35
3.1.2	A Design Space for Focus+Context Interaction Techniques	37
3.1.3	Focus+Context Interaction for Time-series	44
3.1.4	Interacting with Wall-sized Displays	46
3.2	Engineering of Multi-scale Interactive Systems	54
3.2.1	Easing the Development of Multi-scale User Interfaces	55
3.2.2	A User Interface Toolkit for Cluster-Driven Wall-sized Displays	59
4	Perspectives	63
4.1	Interacting with Massive Webs of Data	65
4.2	Human-Computer Interaction and Situation Awareness	66
	Bibliography	71

Appendices	87
A Publications	87
Selected Publications (2006-2011)	91
Fresnel: A Browser-Independent Presentation Vocabulary for RDF	91
Pointing and Beyond: an Operationalization and Preliminary Evaluation of Multi- scale Searching	106
DynaSpot: Speed-Dependent Area Cursor	116
Topology-Aware Navigation in Large Networks	126
Representation-Independent In-Place Magnification with Sigma Lenses	136
High-Precision Magnification Lenses	149
Exploratory Analysis of Time-Series with ChronoLenses	159
Mid-air Pan-and-Zoom on Wall-sized Displays	169
Rapid Development of User Interfaces on Cluster-Driven Wall Displays with jBricks	179
B Résumé en Français	185

Introduction

Advances in our ability to acquire, store, query and process data have led to a spectacular increase in the amount of information that systems can collect and analyze. This has had a profound impact in many domains: scientific research, commerce, finance, industrial processes, as well as all activities related to electronic government. Progress in bioinformatics and in the life sciences at large have led to the creation of immense databases, the Human Genome Project being one of the most emblematic initiatives in this domain. Scientific instruments such as telescopes and particle accelerators generate increasingly detailed observations of our environment and enable ever more complex experiments: the ALMA¹ data production rate has been estimated at approximately 180 terabytes per year; that of the LSST² at 1.28 petabytes a year; and that of the LHC³ at 15 petabytes per year. Advances in high-performance computing enable simulations of increasing complexity, that are being fed, and produce, large quantities of data. Other domains generating large datasets include open data initiatives such as those launched by many governments, digital preservation initiatives such as that of the US Library of Congress and the Bibliothèque Nationale de France, information about customers and their transactions stored by businesses and insurance companies, user-generated content such as blog entries, pictures and videos stored by people on social networks and other Web sites.

In addition to these data sources and providers, recent and emerging technologies such as Web services and the *Web of Data* at large [BL09], ranging from microformats to ontologies on the Semantic Web, have the potential to revolutionize (again) data-driven activities in many domains, by making information accessible to machines as semistructured data [ABS99] that eventually becomes actionable knowledge⁴. Thanks to those technologies, heterogeneous, and possibly autonomous, systems can exchange information, infer new data using reasoning engines and knowledge encoded in ontologies, cross multiple data sources with the ability to resolve ambiguities and conflicts between them. The resulting datasets are often very large, and can be made even larger and more useful by interlinking them, as exemplified by the Linked Data initiative [HB11] (Figure 1.1).

One term has recently arisen to denote the many challenges and research questions posed by the management and effective use of those massive datasets: *Big Data*. While *Big Data* has clearly become a buzzword, those problems and challenges are real, and span numerous fields of computer science research: data and knowledge bases, communication networks, security and trust, data mining, data processing, as well as human-computer interaction.

The goal of human-computer interaction (HCI) can be broadly stated as trying to make computers easier to use while augmenting peoples' capabilities, enabling them to deal with more complex problems, larger datasets, as efficiently as possible, in single-user or cooperative work

¹Atacama Large Millimeter/submillimeter Array, <http://almaobservatory.org>

²Large Synoptic Survey Telescope, <http://www.lsst.org>

³Large Hadron Collider, <http://lhc.web.cern.ch>

⁴Throughout this dissertation we try to employ the terms *data*, *information* and *knowledge* according Ackoff's definition of data, information, knowledge, understanding and wisdom [Ack89]: data are the raw facts, information is data processed to be useful, that helps answer *who/what/where/when* questions; knowledge is coherent information that helps answer *how* questions

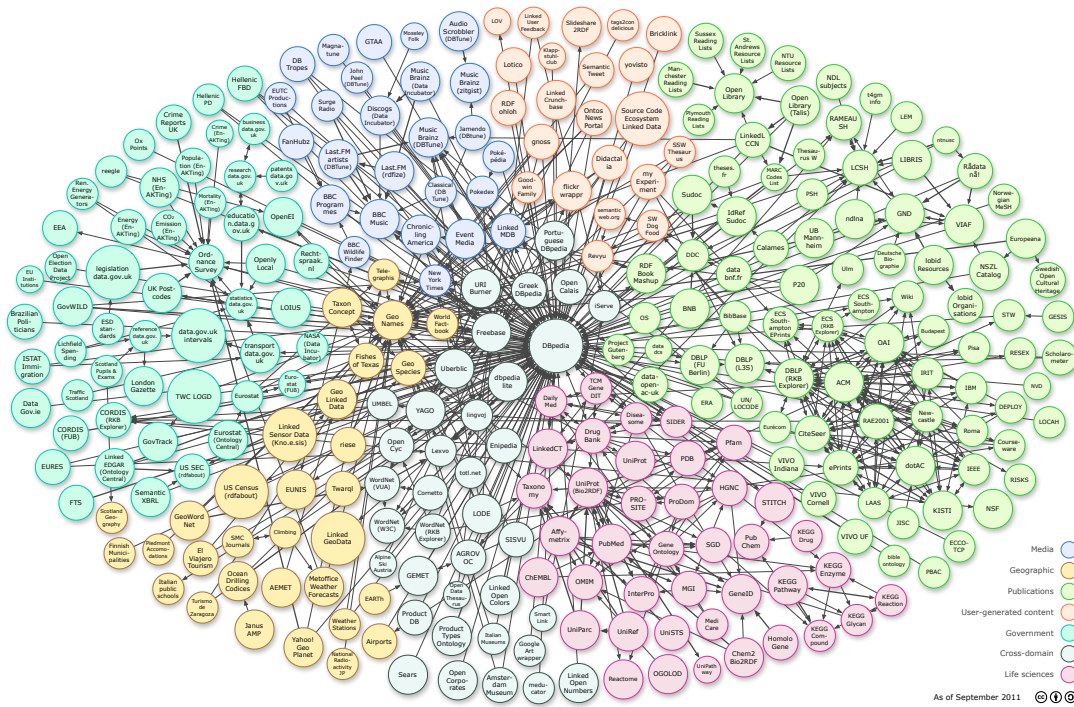


Figure 1.1 : The Linking Open Data cloud diagram (in September 2011): 295 interlinked datasets from varied domains, including e-government, geography, sciences and the media, for a total of more than 25 billion statements.

contexts. A more formal description is that HCI is about designing systems that lower the barrier between the humans' cognitive model of what they want to accomplish and the computer's understanding of the user's task. HCI is concerned with the design, implementation and evaluation of computing systems that humans interact with. It is a highly multidisciplinary field of research, involving experts in computer science, cognitive psychology, design, engineering, ethnography, human factors and sociology. In the more specific context set above, HCI research relates to the design, development and evaluation of interaction and visualization techniques that can help users better comprehend and manipulate large, semistructured datasets. Users can be mere consumers, trying to make sense of the information and to extract knowledge and insights from the data; or they can be producers of those data, creating, structuring, transforming and publishing them for consumption by others; or they can be both. In any case, users are faced with large-to-massive datasets, organized according to more or less complex structures, that can be interlinked as illustrated in Figure 1.1. No matter how elaborate data acquisition, processing and storage pipelines are, the data are produced by users, and eventually get consumed by users in one way or another.

My research is based on the conviction that user interfaces, and more particularly graphical user interfaces, when providing relevant visualizations of the data and their structure coupled

with innovative interaction techniques to navigate in them, can be of great help to users, and thus play an important role in the research and development of computing systems for the management and analysis of massive, semistructured data. My activities have revolved around two main themes, briefly introduced here and presented in more detail in the next two chapters:

- Visual languages and information visualization techniques to help users better make sense of, and manipulate, semistructured datasets.
- The design, implementation and evaluation of multi-scale interaction techniques for navigating large datasets.

1.1 Visualization and Transformation of Semistructured Data

Languages for describing semistructured data, from XML to RDF, OWL and SKOS⁵, enable users to organize their data, but most importantly, they make those data accessible to machines: beyond structure, they provide machine-processable meaning to the data, through domain-specific vocabularies or *ontologies*, and significantly ease their interlinking and merging.

The original Web was, and still is, mostly about serving Web pages to human readers. It is now often called the *Web of documents*, to contrast it with the *Web of data* enabled by the above languages. The Web of data is about sharing information from different sources that can be read automatically, and to some extent reasoned upon, by machines. Consequently, the underlying languages, XML included, are designed to facilitate machine interpretability; the raw data, while encoded using conventional character sets such as UTF-8 and viewable in text editors, is not aimed at being read *as is* by users, as opposed to HTML pages typically served to human readers on the Web of documents.

While most end-users of the Web of data will never see a single line of XML or RDF code, several categories of users are faced with the raw data and have to make sense of it:

- domain experts that create the vocabularies and ontologies that give machine-processable meaning to the data, and populate them with actual data;
- software developers that write programs to query and manipulate the data to, e.g., correlate different sources, infer new data, and eventually transform the resulting data structures to a target vocabulary, such as HTML, for human consumption.

While conducting my PhD from 2000 to 2002 at Xerox Research Centre Europe⁶ and INRIA Rhône-Alpes⁷, I worked on the design and implementation of a visual programming language, VXT, for the specification of XML document structure transformations [PVDQ01, PVD01].

⁵The reader is referred to <http://www.w3.org/standards/semanticweb/> for a description of how these and other technologies build upon, and complement, each other.

⁶<http://www.xrce.xerox.com>

⁷<http://www.inria.fr>

The goal of this language was to ease the development of XML transformations by taking advantage of the multi-dimensional nature of visual languages to explicitly represent the hierarchical structure of the data, deemed an important piece of information in itself, that even relatively high-level languages such as XSLT [W3C99] were only partially conveying due (in part) to their uni-dimensional nature. While a significant part of my work focused on the formal definition of the syntax and semantics of the language and on proving some of its properties (completeness of transformations, well-formedness of productions) [Pie02a, VDP01], I quickly got interested in the human-computer interaction issues that arose while designing a development environment for that visual programming language. Visual languages can be very powerful tools, and some perform very successfully, such as LabView [Pow11]. But they require careful visual design and interaction design to really take advantage of their potential and avoid common pitfalls [GP96]. Problems of scalability are among the main issues encountered when designing visual languages [BBB⁺95]. This was particularly true for VXT, that had to visually represent potentially large and complex hierarchical data structures, the associated grammars (Document Type Definitions, XML Schemas), and transformation rules à la XSLT all using a visual representation paradigm. This led me to survey the field of information visualization, and eventually adopt techniques for tree representations and multi-scale navigation in large information spaces for the purpose of representing, and navigating in, large hierarchical data structures using a zoomable user interface, that enables smooth panning and zooming in large 2D information spaces possibly coupled with semantic zooming capabilities [PF93].

A 4-month internship in the World Wide Web Consortium⁸ (W3C) team at MIT during the fall of 2001 gave me the opportunity to work on a related problem: that of providing users with an interactive visual representation of RDF data. The structure of those data, i.e., directed labeled graphs, is poorly conveyed by textual serializations. While the latter are the primary mean of representing and exchanging information between software agents, they are non-optimal when the information is exchanged between computers and humans. Visual representations can help, but introduce their own problems, and I investigated possible solutions for some of them. This work, that I continued during my post-doc in the Decentralized Information Group⁹ at MIT in 2003, and the subsequent collaboration on the Fresnel language with the Simile project¹⁰ in 2005-2006, is detailed in Chapter 2, along with other collaborative projects conducted in recent years, in which my contribution lied essentially in the design and implementation of semistructured data visualizations for domain-specific applications.

1.2 Multi-scale Navigation in Large Datasets

After my post-doc and a year spent in the industry working on Web-based content management systems, I joined the In-Situ project-team¹¹ at INRIA Saclay as a *Chargé de Recherche* in the fall of 2004. My interest in human-computer interaction, and more particularly in inter-

⁸<http://www.w3.org>

⁹<http://dig.csail.mit.edu>

¹⁰<http://simile.mit.edu>

¹¹<http://insitu.lri.fr>

action techniques and information visualization, had grown over the years, and I was willing to conduct research in, rather than just apply results from, this field. In-Situ provided me with excellent conditions to achieve this transition.

My work in the field of human-computer interaction has essentially focused on the design, implementation and evaluation of interaction techniques for navigating large information spaces, containing visual representations of structured datasets or other types of graphics: large imagery, geographical information systems, digital libraries. Providing means to efficiently navigate such spaces, at different levels of detail, is a general question that has received a lot of attention over the years [CKB08]. But as datasets grow in size, sometimes exponentially, novel, more efficient techniques, better adapted to their context of use [ABM05], have to be designed and validated.

I first worked on a method for operationalizing multi-scale search that enables a more rigorous evaluation of interaction techniques that support this type of task, including pan & zoom, overview + detail and focus + context interfaces. I then proposed several enhancements to the focus + context interface scheme, that smoothly integrates a detailed, magnified region of interest into the surrounding context. I studied alternatives to spatial distortion to achieve smooth transitions between the focus and context regions based on the use of translucence and dynamic adaptation to user input, described a method to render magnification lenses that applies to arbitrary types of graphics, and investigated solutions to the problem of quantization that arises when manipulating lenses set to high magnification factors.

Multi-scale visualization is also the central theme of the ongoing WILD project (Wall-sized Interaction with Large Datasets), in which we study the use of ultra-high-resolution, wall-sized displays for the visualization of massive scientific datasets. Research on WILD, a 131-million-pixel display, is organized around three main themes: the design and evaluation of interaction techniques adapted to this type of platform; the user-centered design of scientific visualization applications involving scientists from various disciplines such as astronomy and neurosciences; the design and development of user interface toolkits that facilitate the rapid prototyping of novel interaction and visualization techniques, hiding the complexity that is inherent in this type of environment, where displays are driven by clusters of computers and where interaction involves multiple heterogeneous input devices.

These contributions to the field of human-computer interaction, as well as related ones on the topic of pointing facilitation and other desktop interaction techniques, are detailed in Chapter 3.

Appendix A contains selected publications that are representative of my work in both themes.

Techniques for the Visualization of Semistructured Data

The goal of the Semantic Web is to enable machines to more easily exchange, merge and reuse datasets by creating a *Web of data* where the data has clearly-defined, machine-processable semantics [SBLH06].

The Semantic Web is based on a set of languages and technologies that build on top of one another. Uniform Resource Identifiers (URI) [BLFM05] and the more recent Internationalized Resource Identifiers (IRI) [DS05] provide an identification scheme for Web resources; the XML markup language provides a syntax for representing semistructured data. On top of these, the Resource Description Framework (RDF) is the foundational layer for structuring the data, publishing it on the Web, and easily interlinking datasets [MM04]. Additional layers build on RDF, including RDF Schema and the Web Ontology Language (OWL) [MvH04] to organize the data and give it machine-processable semantics, query languages such as SPARQL [PS05], and inference mechanisms that reason over the data using rules and the semantics captured in ontologies.

The languages of the original Web, such as HTML and SVG (Scalable Vector Graphics) [W3C07], are meant to facilitate the exchange of documents between people, who are the primary consumers of those resources. The above languages, including XML, RDF and OWL, are designed to facilitate machine interpretability of information and do not define a visual presentation model since human readability is not among their primary goals. However, Semantic Web data is, at least partially, created and manipulated by people: software developers that write programs to query and manipulate the data; domain experts who do not necessarily have advanced skills in computer science but still have to understand the data. The latter have to be easy enough to create, read, modify and interlink for this Web of data to gain momentum and be successful. Downward in the data processing pipeline, end-users eventually make use of the data, or a subset of it, presented to them in some form [DLK⁺10]. Those data have to be displayed in a human-friendly way, that raw textual serializations do not allow, requiring solutions for data restructuring and transformation to target formats such as HTML pages, PDF documents, interactive visualization components (maps, charts, etc.) or other presentation means.

Section 2.1 describes the work on IsaViz, a multi-scale graphical editor that represents RDF graphs as node-link diagrams and enables more relevant representations than the ubiquitous conceptual graph representation through Graph Style Sheets. Section 2.2 describes the Fresnel RDF presentation vocabulary that addresses the latter need by enabling the high-level, declarative specification of presentation rules of RDF datasets in a manner that is independent of the representation paradigm. Finally, Section 2.3 gives an overview of my subsequent work on interactive graph visualization, that led to multiple collaborations and applications.

2.1 Graphical Transformation and Representation of Semantic Web Data

Statements, made of a *subject*, a *predicate* and an *object*, are the basic elements, sometimes called *triples*, that constitute RDF models. The subject and object are connected together through the predicate, expressed by a *property* that characterizes the relationship between them. Sets of statements form RDF *graphs*, whose nodes are the subjects and objects of those statements, and edges the predicates that link them. The relationship expressed in a statement is always directed from the subject to the object. The RDF data model is thus that of a structured, labeled graph [KC04].

Multiple textual formats have been defined for the serialization, exchange and raw editing of RDF data. RDF/XML [Bec04], perhaps the most well-known of them, encodes RDF graphs as XML trees. This format benefits from all the XML machinery and tools that facilitate data interchange, but it is totally illegible for most users. Not only is the graph serialized using a relatively complex and verbose syntax, but the XML tree structure the original graph is projected on has absolutely no meaning, creating a level of indirection forced on developers and applications by the constraints of the XML data structure. Other, more user-friendly formats exist, such as Turtle [BBLP11] and Notation 3 (N3) [BL05]. The latter is often used by software developers who manipulate raw RDF data directly. However, like all other textual serializations, this format has one weakness due to its uni-dimensional nature: it cannot explicitly represent the graph structure of RDF models, as shown in Figure 2.1.

The way the data are structured on the Semantic Web depends a lot on the RDF vocabularies used to describe those data. Some vocabularies generate acyclic, monotonous and shallow tree-like structures, while others generate complex directed graphs containing widely different substructures. The organization of the data can be a very important piece of information in itself, beyond the values of the raw data items. The additional structure created by the links that interconnect independent datasets can also be an important element, that users have to be able to understand, especially when different vocabularies are used in the same graph to describe multiple aspects of a resource, or when resources from different domains are linked with one another [PL09].

2.1.1 Multi-scale Visualization of RDF Graphs

Hierarchical and graph structures can be more easily understood when visualized using graphical representation techniques. Graph visualization techniques [HMM00] represent the structure explicitly, and can lower users' cognitive load significantly when their tasks involve getting an understanding of how the data is structured, both globally and locally. As an example, compare Figure 2.1 with Figure 2.3-a, both representing the same RDF data.

However, graph drawing techniques alone do not scale to graphs that contain more than a few hundred nodes and edges, no matter the layout algorithm employed. Larger graphs require the

```

<http://www.daml.org/cgi-bin/airport?BOS>
  a airport:Airport ;
  airport:iataCode "BOS" ;
  airport:name "Logan Airport" .

_:b1 a foaf:Person ;
  airport:nearestAirport <http://www.daml.org/cgi-bin/airport?BOS> ;
  foaf:surname "Swick" ;
  foaf:givenname "Ralph" .

[] a foaf:Person ;
  airport:nearestAirport <http://www.daml.org/cgi-bin/airport?CDG> ;
  foaf:depiction <http://www.lri.fr/~pietriga/id.jpg> ;
  foaf:homepage <http://www.lri.fr/~pietriga/> ;
  foaf:knows _:b1 ;
  foaf:knows [ a foaf:Person ;
    airport:nearestAirport <http://www.daml.org/cgi-bin/airport?BOS> ;
    foaf:knows _:b1 ;
    foaf:surname "Lee" ;
    foaf:givenname "Ryan" ] ;
  foaf:knows [ a foaf:Person ;
    airport:nearestAirport <http://www.daml.org/cgi-bin/airport?BOS> ;
    foaf:surname "Prud'hommeaux" ;
    foaf:givenname "Eric" ] ;
  foaf:surname "Pietriga" ;
  foaf:givenname "Emmanuel" ;
  foaf:phone <tel:+33-1-69153466> ;
  foaf:workplaceHomepage <http://www.inria.fr> .

<http://www.daml.org/cgi-bin/airport?CDG>
  a airport:Airport ;
  airport:iataCode "CDG" ;
  airport:name "Charles de Gaulle Airport" .

```

Figure 2.1 : N3 serialization of an RDF graph describing people, their friends (FOAF vocabulary) and the airports nearest the cities where they live (DAML Airport vocabulary).

visualization software to provide users with interactive navigation capabilities that will enable them to move from an overview of the graph to zoomed-in, detailed renderings of particular regions of interest.

My work on IsaViz [Pie02b] was one of the first attempts at creating a visual interactive authoring tool for RDF models (Figure 2.2). IsaViz was based on a zoomable user interface component that I had started developing during my PhD as part of my work on the development environment for the VXT visual programming language (Section 1.1). This component eventually grew into a full-fledged post-WIMP [BL00] user interface toolkit now called ZVTM [Pie05a], that has been, and is still actively, used in many projects, as detailed in the next sections. IsaViz enabled users to load moderately large RDF models (several thousands of triples) and smoothly pan & zoom in the graphical representation of RDF graphs whose layout was computed by Graphviz/dot [EGK⁺01].

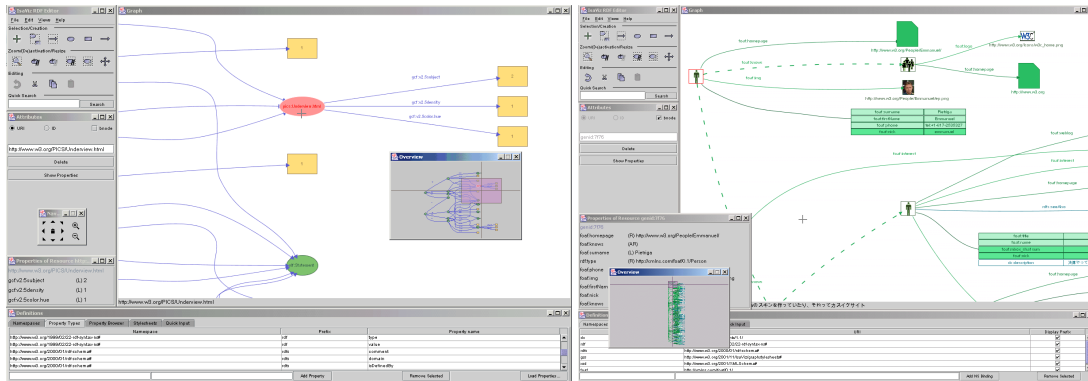


Figure 2.2 : Visual Authoring of RDF graphs with IsaViz.

IsaViz offered editing capabilities, that let users add, edit and remove or deactivate¹ resources and whole statements in a data-model aware manner to make sure that RDF models were always syntactically correct: the editor would not constrain user actions, but would ensure the model was still valid by, e.g., removing dangling edges when a resource had been removed from the graph.

2.1.2 Graph Style Sheets

IsaViz was welcome by the community and got a lot of visibility, being distributed² by W3C. However, it had two major flaws. First, depending on their structure, some RDF models would translate into dense graph layouts with many crossing edges that would clutter the screen and would make the representation illegible (Figure 2.4-a). Second, the representation was that generally used for representing simple RDF graphs [KC04]: all elements were represented in the same manner, independently of the type of resource or predicate, as illustrated in Figure 2.3-a. Resources were represented as URIs in ellipses, literal property values as boxed text, and statements as solid arrows. This representation was acceptable for very simple graphs, but was not optimal for complex graphs, especially those describing resources using multiple vocabularies and ontologies.

Graph Style Sheets (GSS) [Pie03, KKPS03] addresses those two problems by providing graphical styling, filtering and visual data restructuring capabilities in IsaViz. GSS is a style sheet language that makes it possible to transform the default node-link diagrammatic representation through the declarative specification of visibility, layout and styling rules applied to its nodes and arcs. Graph Style Sheets can be considered as graph transformations of a specific kind, although they are conceptually much simpler: Graph Style Sheets are not general graph rewriting systems, in the sense that GSS rules do not replace matched subgraphs with other arbitrary subgraphs; the GSS transformation process is about taking the “default” representation of an RDF graph displayed as a node-link diagram (Figure 2.3-a), and modifying the visual

¹Deactivation of statements is like commenting out lines in the source code of a program or text data file.

²<http://www.w3.org/2001/11/IsaViz/>

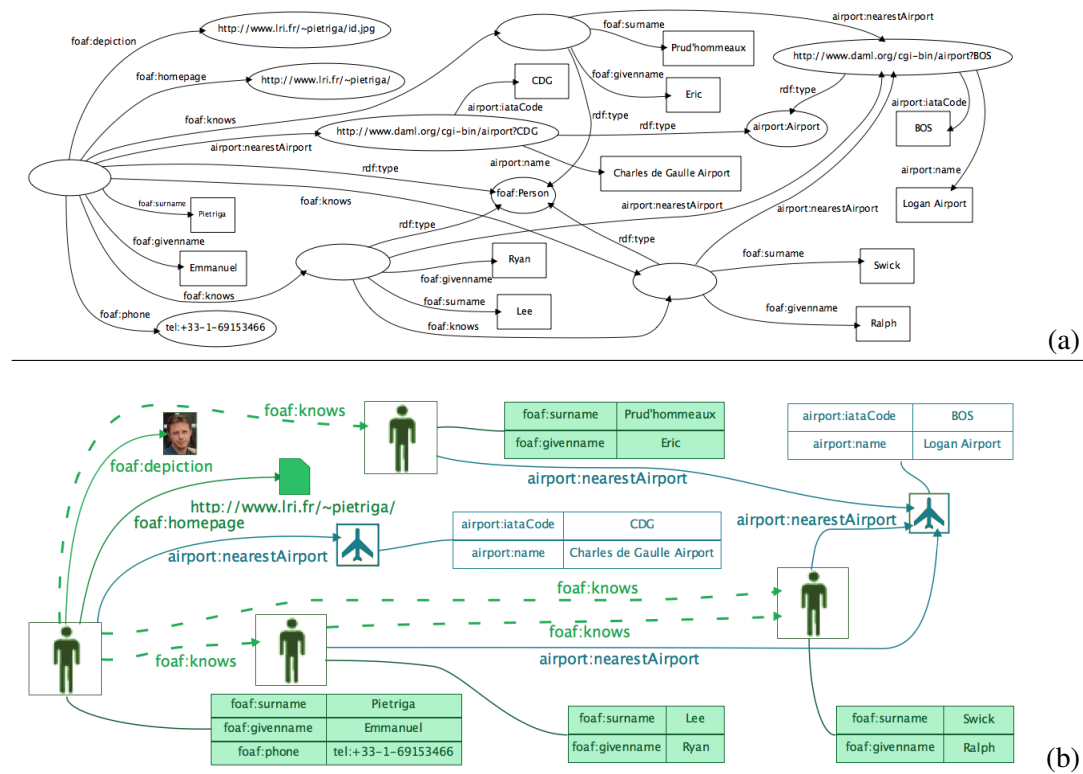


Figure 2.3 : Default (a) and GSS (b) rendering of the same Friend-of-a-Friend model with DAML Airport data as in Figure 2.1.

appearance of its elements (Figure 2.3-b). It is thus conceptually much closer to the process of applying the now ubiquitous CSS style sheets [W3C98] to the default rendering of HTML pages.

In the default node-link diagram representation of RDF graphs that was originally employed in IsaViz, variables such as color, font, shape border thickness and stroke pattern (solid, dotted, dashed, etc.), did not bear any semantics, thus giving significant freedom to customize the visual rendering of elements. GSS rules specify styling instructions, that are modifications made to these variables; content reorganization instructions, such as grouping a set of related properties in a table; and visibility instructions that can be used to hide elements of the graphs considered as irrelevant for a given task backed by the visualization. GSS rules are composed of a *selector* on the left-hand side, associated with a *styling instruction set* on the right-hand side. GSS' execution model is similar to that of CSS, though applied to a directed labeled graph instead of a tree: given the set of rules defined in a style sheet, the GSS processor walks the entire graph, i.e., all nodes and arcs, and evaluates relevant rules on them. If the selector of a rule matches the current node (or arc) in the graph, the corresponding set of styling, visibility and layout instructions is applied to that node (or arc).

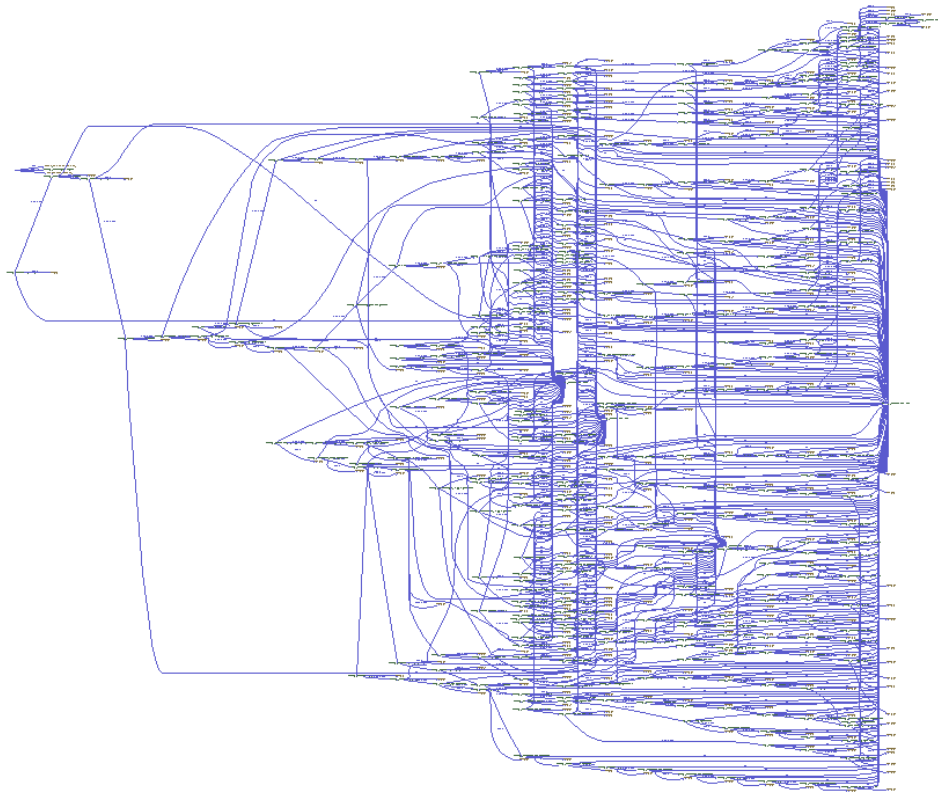
The simplicity with which two RDF graphs can be merged³ means that datasets can easily be interlinked (actually, this does not require an explicit operation beyond making the triples of both datasets available to the client application), and that different users and organizations can describe, or "say things" about the same resources, possibly using different RDF vocabularies. While one could imagine one huge style sheet containing rules for all classes and properties defined in all known ontologies, this approach would not be practical and would cause many problems, as designing a single ontology for all things in the world would. Style sheet designers are typically expected to create one style sheet per RDF vocabulary, though this is not a requirement. As RDF models often make use of several vocabularies to describe resources, GSS features a cascading mechanism similar to that of CSS that makes it possible to apply several style sheets to the same model. Style sheets can easily be merged together, GSS being itself expressed in RDF, and thus benefiting from all the features enabled by the framework. Style sheets will often act on different elements of the graph, but conflicts between rules may arise and are handled through the computation of a specificity metrics for each rule.

In Figure 2.3, the Friend-of-a-Friend (FOAF) RDF vocabulary is used to represent information about people (contact information, current projects, workplace, etc.) and about the social network that links them. The DAML Airport vocabulary provides information about airports nearest to the people involved in the network. Two GSS style sheets are applied to the original model: the first one applies to FOAF elements while the second one applies to DAML Airport elements. The style sheets make it easier to get a general understanding of the graph and to extract information without having to go in the low-level details of the representation. For instance, all nodes and arcs representing class membership information have been removed, but this essential information is still conveyed, as resources are now depicted by icons or shapes representative of each class (Airport, Person, Document). Nodes representing persons have been made larger than other nodes as they represent the central elements of the network. Other styling instructions include using similar color hues for the elements of a given vocabulary (green for FOAF, blue for Airport), and grouping related information for a given resource in tables, e.g., all contact information about a person.

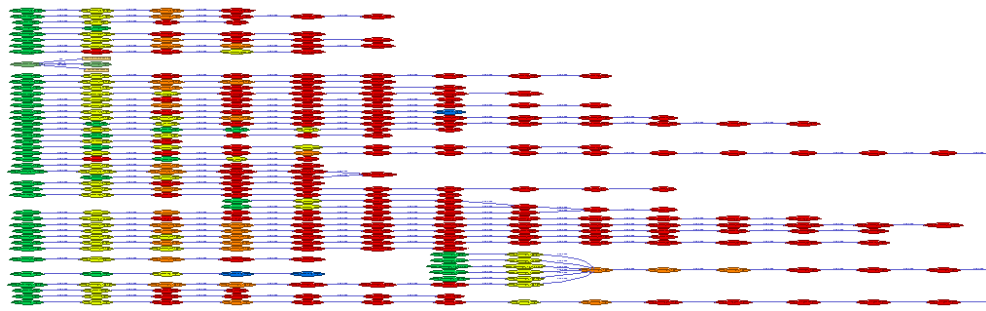
Figure 2.4 illustrates GSS on another dataset. [VPJM05] describe how IsaViz and GSS were extended to create a domain-specific application for the representation of metabolic and regulatory networks.

Significant changes were made to GSS in [Pie06], aimed at enhancing the language, based on user feedback and on our own experience developing style sheets. The most important change was that the selector language, based on RDF's reification mechanism, was eventually replaced by the Fresnel Selector Language [Pie05b] described in Section 2.2, as the original language was powerful but too verbose and too complex to express simple things.

³URIs provide unique identifiers for resources. Merging the two graphs of two RDF datasets is just as simple as putting all triples together in the same "bag".



(a)



(b)

Figure 2.4 : (a) Default – totally illegible – rendering of documents going through the W3C Recommendation track workflow; (b) GSS rendering of the same data, with a very simple set of styling rules: one rule colors nodes according to their status in the workflow, another rule hides the corresponding statements. As a result, the graph structure gets significantly simplified. It then becomes easy to observe several facts about the dataset, that are very difficult to see in the default rendering, e.g., some documents went through many *working draft* (red) iterations before going through the next steps, while others reached the next state more quickly; some actually went back to *working draft* from *candidate recommendation* (yellow); one document was split into six different parts between the *proposed recommendation* (orange) and *candidate recommendation* steps.

2.1.3 Visualization of Populated Ontologies

As Semantic Web technologies are slowly but steadily gaining adoption, more and more datasets are made available in the form of populated ontologies, many of which contain millions of triples. Conventional node-link diagram representations do not scale to such large datasets. Actually, no existing visualization technique scales to such large quantities of triples, and only overviews or summaries of the data can be generated [TXZ⁺05]. In their position paper, schraefel and Karger [sK06] were questioning the value of “*big fat graph*” representations. While it is certainly true that visualizations based on node-link diagrams should not be the default mechanisms for the representation of Semantic Web data, my position is that generic visualizations are still useful and can effectively support tasks such as the exploration of datasets the user is not very familiar with. Carefully designed visualizations can be effective at providing insight into the data, and help users “*answer questions they didn’t know they had*” [Pla04]. Furthermore, depending on the domain, explicitly visualizing the structure by graphically depicting relations can be helpful for some tasks, as the underlying graph structure by itself bears information [MG03] (e.g, social network analysis, or applications in bioinformatics).

In 2010, I worked with Benjamin Bach, now a PhD student co-advised with Jean-Daniel Fekete, on the problem of visualizing populated ontologies using techniques that would scale to datasets one order-of-magnitude larger than what node-link diagrammatic representations enabled.

Two main issues with node-link diagram representations of ontology graphs are their inefficient use of screen real-estate and edge crossings that make dense regions difficult to read. A well-known alternative to node-link diagrams for graph visualization are adjacency matrices [HFM07, ZKB02]. Nodes are represented as rows and columns, and edges as filled cells at the intersection of connected rows and columns. While node-link diagrams are good at showing the structure of relatively small and sparse graphs, adjacency matrices are very effective at showing large (better use of screen real-estate) and dense (no edge crossing) graphs. However, adjacency matrix representations are much less familiar to users than node-link diagrams, and make tasks that involve following paths in the graph more difficult [HFM07], significantly increasing the user’s cognitive load.

We designed and implemented a new ontology representation tool (Figure 2.5), OntoTrix [BLP10, BPLL11], inspired by a hybrid visualization called NodeTrix that was originally applied to social networks [HFM07]. NodeTrix is very efficient at visualizing locally dense but globally sparse networks, representing the overall structure of the network using a node-link diagram and the dense subgraphs that represent communities using matrices: “*Intra-community relationships use the adjacency matrix representation while inter-community relationships use normal links*” [HFM07]. While the graph structure of ontologies might not always share the small-world characteristics of social networks, we believed that such a hybrid representation, combined with appropriate interaction techniques, could be an efficient means to perform exploratory visualization of large ontology instance sets. Indeed, this hybrid visualization pro-

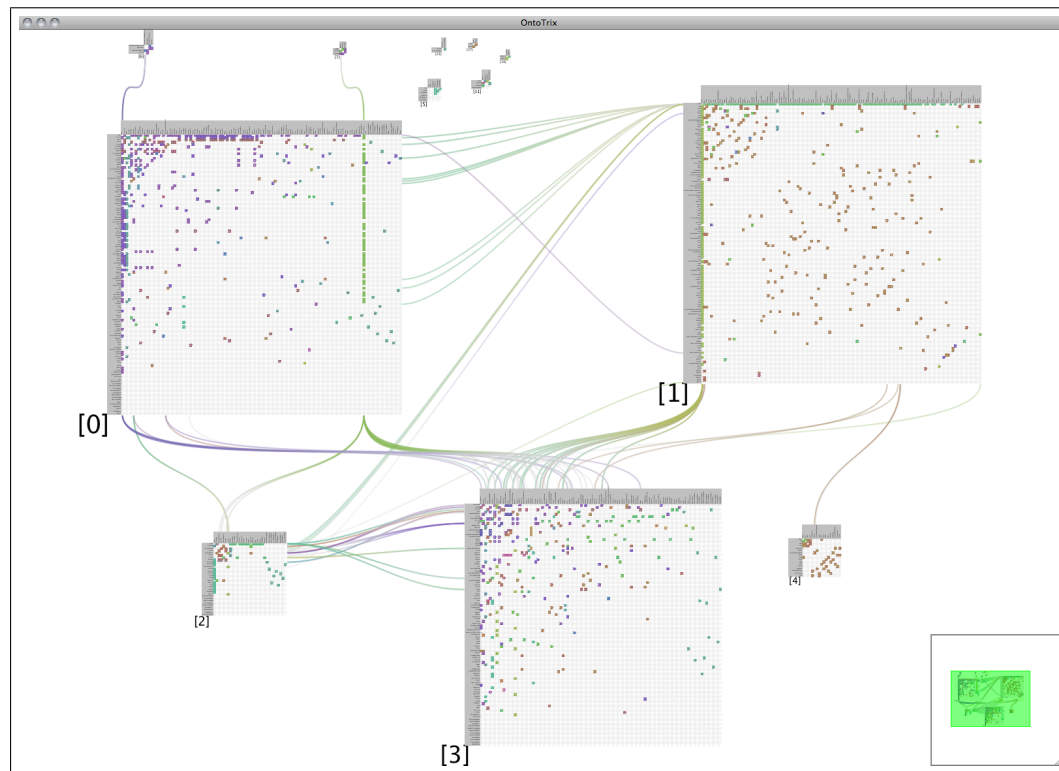


Figure 2.5 : Visualizing 724 instances from 49 classes, and 1 636 object properties (29 definitions) from the NTN ontology with OntoTrix.

duces more compact and legible representations than node-link diagram approaches, as it suffers significantly less from the spaghetti-like effect due to edge crossing problems typically encountered with conventional node-link layouts of non-planar graphs.

We extended NodeTrix to handle the much richer and complex graph structures of populated ontologies (different types of nodes, different types of relations) compared to basic social networks, exploiting ontological knowledge to drive the layout of, and navigation in, the representation of instances and their relations. To provide enhanced task and cognitive support to users, the OntoTrix representation is embedded in a zoomable environment implemented on top of our zoomable user interface toolkit (ZVTM, Section 3.2), and coupled with highly-coordinated [NS00] views of the class and property hierarchies that enable interactive navigation and filtering of instance data. Using OntoTrix, we managed to get legible visualizations of populated ontologies such as the Wine ontology⁴ (4,547 statements), and up to 23,665 statements in the SC ontology⁵, something that would have been extremely challenging (at best) with a conventional node-link representation.

⁴<http://www.w3.org/TR/2003/CR-owl-guide-20030818/wine>

⁵<http://www.mindswap.org/ontologies/SC.owl>

2.2 A Presentation Vocabulary for the Web of Data

The target audience for IsaViz was mainly domain experts and software developers who have an interest in visualizing the structure of the data they manipulate at a low-level of abstraction. Other audiences, closer to *end-users* in the sense that they are not programmers or data architects, have used it in conjunction with GSS to visualize and manipulate data in a particular domain, such as biologists with *Metabolic IsaViz* [VPJM05], but these presently remain exceptions to the norm. However, software developers and domain experts are not the only consumers of Semantic Web data. As discussed earlier, the data, or a subset of it according to Shneiderman's visual information seeking mantra "*Overview first, zoom and filter, then details on demand*" [Shn96], must eventually be presented to end-users. Those data have to be displayed in a human-friendly way, requiring solutions for data restructuring and transformation to target formats such as HTML pages [Sim05, BLCC⁺06] (Figure 2.6-left) possibly containing interactive visualization components such as maps and charts [HMK05], sometimes produced as mashups [AGM08, ZR08]. They can also be output as PDF documents and similar formats for more static content; or to rich clients featuring advanced graphics capabilities [QHK03, sSO⁺05] for dynamic content or when providing a highly-interactive user experience (Figure 2.6-right).

In any of these cases, the problem is the same: presenting content primarily intended for machine consumption in a human-readable way. Semantic Web browsers and related applications offer solutions that differ but in the end address the same two high-level issues, no matter the underlying representation paradigm: specifying (i) *what* information contained in RDF models should be presented (content selection) and (ii) *how* this information should be presented (content formatting and styling). However, each tool currently relies on its own *ad hoc* mechanisms and vocabulary for specifying RDF data *presentation knowledge*, making it difficult to share and reuse such knowledge across applications.

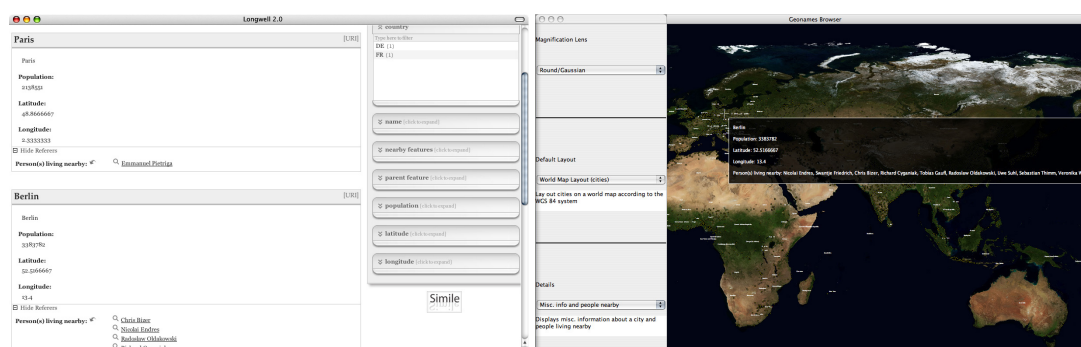


Figure 2.6 : Two different applications using the same Fresnel lenses and formats to display information from `geonames.org`: (left) the Web-based Longwell faceted browser generates HTML+CSS pages that can be served to any Web browser; (right) a prototype multi-scale geographical information system implemented in Java2D+Swing, overlays the data on top of NASA's Blue Marble Next Generation world map (dimensions: 86 400 x 43 200 pixels).

From 2005 to 2006, I worked in collaboration with researchers from MIT and from Freie Universität Berlin on the design and implementation of Fresnel⁶, a browser-independent vocabulary of core RDF display concepts [PBKL06].

Our goal with Fresnel was to provide a vocabulary to encode information about how to present Semantic Web content to users (i.e., *what* content to show, and *how* to show it) as presentation knowledge that can be exchanged and reused between browsers and other visualization tools. We identified a set of core presentation concepts that were meaningful across applications and that formed the core modules of Fresnel. One of the design goals of these modules was to make them easy to learn and use, but also easy to implement in order to promote their adoption by many applications. This effort has been relatively successful, the vocabulary now being used in several applications and *linked data* browsers, and cited by more than one hundred and thirty scholarly articles five years after its publication.

Fresnel is itself an RDF vocabulary, described by an OWL ontology. Fresnel presentation knowledge is thus expressed declaratively in RDF and relies on two foundational concepts: *lenses* and *formats*. Lenses specify which properties of RDF resources are shown and how these properties are ordered, while formats indicate how to format content selected by lenses and optionally generate additional static content and hooks in the form of CSS class names that can be used to style the output through external CSS style sheets.

As GSS styling rules, both Fresnel lenses and formats apply to specific RDF resources and properties only, as defined by their *domain*. Domains are selection expressions that can express constraints as basic as class membership, or possibly much more elaborate ones written with either the SPARQL query language for RDF [PS05], or the Fresnel Selector Language (FSL) [Pie05b], that offers developers a more XPath-like⁷ way of expressing semistructured data pattern matching constraints [CD99].

Fresnel lenses specify what properties of resources to display. Several lenses can apply to the same resource, and ask for different property sets to be displayed. For instance, a *summary* lens that applies to a `foaf:Person` would only show her name, while a *details* lens would also show a depiction of that person, if available, as well as, e.g., contact information and/or links to her friends on the social network. It is then up to the application, depending on the context, to choose the most appropriate lens to render a given resource. In addition, lenses provide constructs to handle the potential irregularity of RDF data stemming from the fact that different authors and organizations might use similar terms coming from different vocabularies to make equivalent statements. Sets of properties considered as similar for a given purpose can be declared as such. For instance, a lens could declare `foaf:depiction`, `foaf:img` and `p3p:image`⁸

⁶<http://www.w3.org/2005/04/fresnel-info/manual>

⁷XPath has proven to be a well-adapted selector language for XSLT transformation rules. FSL thus adopts a similar syntax, but is a genuine path language that works at the RDF graph level, not an adaptation of XPath that would apply to RDF/XML tree projections of RDF graphs. Nevertheless, the syntax and data model are relatively similar, making it straightforward for somebody knowledgeable about XPath to write FSL pattern matching expressions.

⁸Platform for Privacy Preferences Project. <http://www.w3.org/P3P>

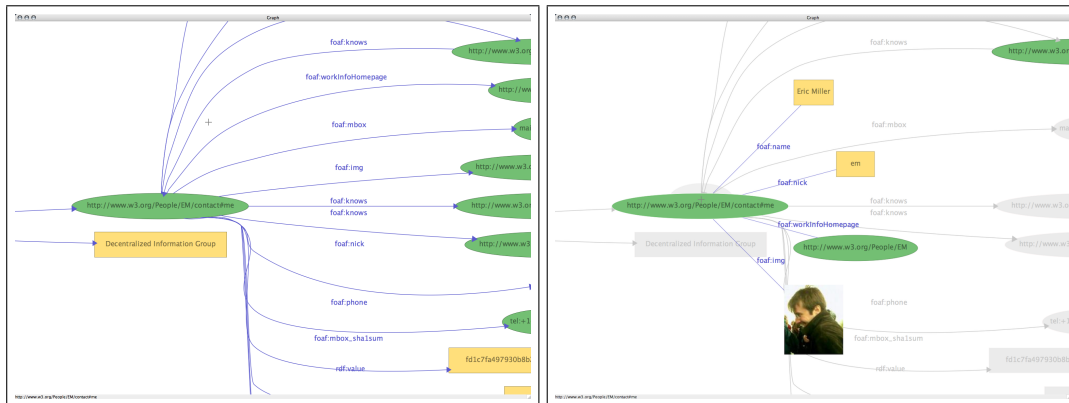


Figure 2.7 : Fresnel in IsaViz: (left) default rendering of a region of an RDF model containing a `foaf:Person`. At this level of magnification, only a few of the many property values associated with the resource are visible. Users need to navigate in the graph in order to get to the values of properties, which can be cumbersome. Alternatively, users can select a Fresnel lens from the list of available lenses loaded in IsaViz. The selected lens is then tied to the mouse cursor, and when the lens hovers over a resource that matches its domain, the resource’s visual appearance gets modified according to the lens and associated format(s). Resources that match the selected lens’ domain are made visually prominent by rendering all other nodes and all arcs using shades of gray with minimum contrast. When the lens hovers over a resource, properties selected by the lens are temporarily rendered with highly-contrasted vivid colors and brought within the current view, closer to the main resource and reordered clockwise according to the ordering of properties in the lens definition. Property values revert back to their original state when the lens moves away from the resource.

as similar. The Fresnel engine would first look for a `foaf:depiction` to visually represent a `foaf:Person`. If it failed to find this property, it would then look for a `foaf:img`, and then for a `p3p:image` if necessary. Similarly, all values of properties considered equivalent can be merged together in a single list that can later be formatted as a whole. Such knowledge can also be represented through ontology mapping mechanisms, but Fresnel provides these constructs as the ontology layer should not be made a requirement of the Fresnel presentation process, which should be as lightweight as possible and should not rely on possibly computationally-expensive inference mechanisms. Furthermore, what is formally considered at the ontological level as equivalent or not might be or might not be considered as equivalent for the purpose of presentation in a given context.

The presentation of property values is not limited to a single level, and (possibly recursive) calls to lenses can be made to display details about the value of a property. Lenses used in this context are referred to as *sublenses*. Fresnel defines a closure mechanism to prevent infinite loops caused by recursive calls that would otherwise occur when lenses call themselves as sublenses for displaying property values.

The default layout of selected information items is highly dependent on the browser's representation paradigm. A Web-based browser such as Longwell (Figure 2.6-left) will typically adopt the nested box layout model defined by HTML+CSS; graph visualization tools such as IsaViz will typically represent the data as node-link diagrams (Figure 2.7-left); rich clients such as Haystack [QHK03] and the prototype depicted in Figure 2.6-right will use different paradigms depending on the widget used to represent the information, ranging from basic WIMP widgets such as buttons and checkboxes, to calendars and maps. But the rendering can usually be customized by associating high-level formatting and styling instructions with elements of the representation.

Fresnel formats give high-level directives regarding how the resource and its properties should be presented. For instance, formats can be used to set a property's label, or to specify how to separate the multiple values of a given property or set of merged properties, for instance using commas. Formats can also give instructions regarding how to render values: as clickable links (email addresses), as bitmap images, etc. For lower-level (graphical) formatting and styling, CSS class names can be associated with the various elements being formatted. These names appear in the output document and can be used to style the output by authoring and referencing CSS style sheets that use rules with the same class names as selectors. We chose to re-use existing languages for this level of formatting, as languages such as CSS fulfill that requirement well, and redefining yet another styling language would have put an unnecessary load on both Fresnel engine implementors and Fresnel presentation developers, without bringing any clear benefit.

Obviously, Fresnel implementations will apply those formatting instructions differently, depending on the underlying representation paradigm. For instance, IsaViz uses Fresnel lenses not as a general indication about how to organize information, but as an interaction technique to navigate large node-link diagrams. As illustrated in Figure 2.7, Fresnel lenses are used to temporarily bring inside the current viewport the property values of the resource hovered by the lens that are not visible in that viewport but are declared by the lens as properties to display, and that would otherwise have required the user to perform tedious pan & zoom navigation actions.

Fresnel - A Browser-Independent Presentation Vocabulary for RDF [PBKL06], available in Appendix A, gives more information about Fresnel. As mentioned earlier, several Fresnel engines have been implemented, by project members as well as third parties. One implementation, called JFresnel⁹, was developed at INRIA. It is used as the Fresnel engine in IsaViz, in the tool depicted in Figure 2.6-right, as well as in the ANR RNTL Project WebContent¹⁰ that ran from 2005 to 2009 and gave us the resources to develop a significant part of the engine. This implementation was later used by HP Labs, by members of project SELE at Masaryk University, and by the Simile Longwell browser at MIT, which used JFresnel to provide FSL support

⁹<http://jffresnel.gforge.inria.fr>

¹⁰<http://www.webcontent.fr>

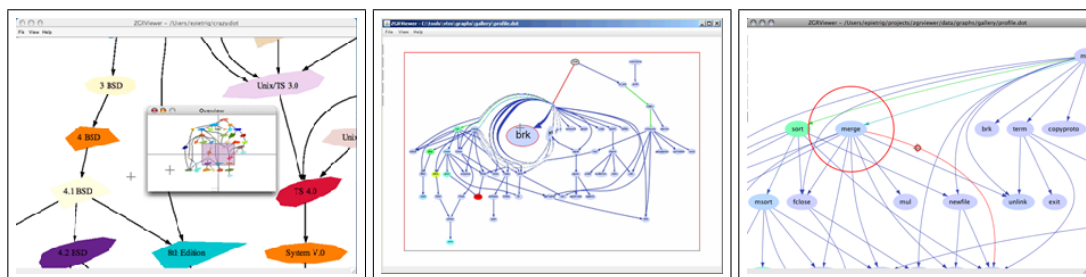


Figure 2.8 : Multi-scale visualization of graph layouts generated by GraphViz with ZGRViewer.

in its own engine. Work on FSL also gave me the opportunity to work with the bioinformatics group at Université Paris-Sud, where it was extended to model queries for the selection of biological data sources and tools [CFP06].

2.3 Graph Visualization

Graph visualization [HMM00] is a rich and very active area of research, with its own yearly conference, and many papers published about that topic at conferences such as IEEE InfoVis. It has many domains of applications beyond the visualization of Semantic Web data, including visual analytics, software engineering, document workflows, social networks, communication networks, systems biology, and industrial processes.

2.3.1 Multi-scale Visualization of Large Graphs

From 2003 onward, while working on Graph Style Sheets, I realized that there was a strong demand for interactive graph visualization tools that would scale to large graphs and enable smooth navigation in the structure. While there were already some toolkits and applications written for that purpose, there was no tool that would enable the interactive visualization of graph drawings generated by layout algorithms from the well-known GraphViz package¹¹ developed by AT&T, beyond basic applications such as *dotty* that could hardly handle graphs composed of more than a few dozens of nodes.

The code developed for visualizing RDF graphs in IsaViz was using GraphViz to compute the initial layout of the RDF graph and on ZVTM to display the resulting SVG vector graphics document (Section 3.2). But that code was actually modular enough that the GraphViz SVG output parsing code and ZVTM scene graph construction code could be extracted and re-used to create a generic tool for the visualization of arbitrary graph layouts generated by any of the GraphViz layout programs.

¹¹<http://www.graphviz.org>

I thus started to work on a new lightweight tool, called ZGRViewer (for Zooming GRaphviz Viewer) that would enable users to smoothly pan & zoom in large graphs generated by GraphViz programs, a capability that existing software viewers lacked. Beyond smooth zooming capabilities, the tool initially offered only a few features: an interactive bird's eye view for overview+detail navigation [CKB08] and the possibility to run as an applet. This effort eventually became a standalone open-source project¹² with a relatively wide audience¹³ spanning many domains, due in part to the widespread use of the GraphViz software package itself. Novel features were added as the audience and use cases grew, including topology-related operations and import/export capabilities.

While the development of ZGRViewer has never been a research project *per se*, it has served as a showcase for my research activities, demonstrating the capabilities of the ZVTM toolkit [Pie05a] (Section 3.2), and acting as a testbed for the design of several multi-scale interaction techniques [PAB07, PA08, PBA10], techniques specifically aimed at navigating network structures [MCH⁺09], and a technique for the selection of small interface components [CLP09]. See sections 2.3.2 and 3.1 for additional information.

Most importantly, and possibly a consequence of the above, it has given me several opportunities to directly collaborate on research projects with scientists from other fields. First with *Universidad Politécnica de Valencia* on the integration of graph visualization capabilities in a bioinformatics tool for functional genomics research [CGGG⁺05]; and now on social network visualization (ANR project Multi-Level Social Networks¹⁴), and on another ongoing project with the *Japan Advanced Institute of Science and Technology* related to the use of formal methods for the modeling and simulation of the behavior of, and interactions between, biological entities such as the λ phage virus.

2.3.2 Interactive Navigation in Large Graphs

To navigate in large graphs, interactive visualization tools including ZGRViewer and IsaViz originally provided generic techniques only: pan & zoom, and sometimes a bird's eye view. However, for large graphs, but also for large roadmaps and other representations of networks with geo-coordinates such as airline route maps or subway maps, some important tasks related to the graph or network's topology are not efficiently supported by these techniques. For instance, using Google Maps, exploring a route often involves panning over long portions of a highway with no exits. Zooming out or using a bird's eye view is possible, but some highway exits are difficult to distinguish from roads passing over or under the highway, and an exit can be missed. The same problem arises in graph visualization tools where nodes are connected

¹²<http://zvtm.sf.net/zgrviewer.html>

¹³A Google search returns 9,640 results. ZGRViewer has been downloaded 35,586 times from `sf.net`. This does not account for Maven dependency downloads, SVN checkouts, or for users of the applet version served on many third-party Web pages. Statistics collected on February 21th, 2012.

¹⁴Collaborative project involving Adis (economics, *Université Paris-Sud*), Irisso (social sciences, *Université Paris-Dauphine*), In-Situ (situated interaction, INRIA, CNRS & *Université Paris-Sud*) and BasicLead (SME). Principal Investigator for partner INRIA: Emmanuel Pietriga.

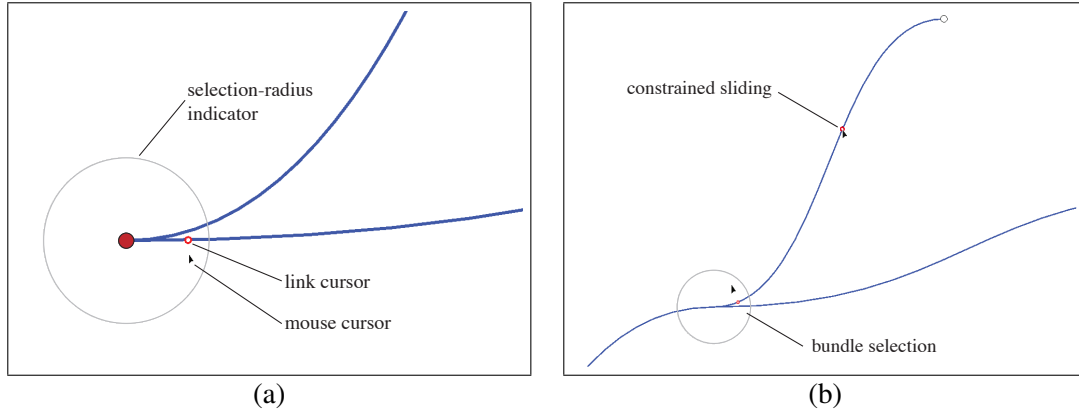


Figure 2.9 : Link Sliding technique: (a) When the technique is engaged in the vicinity of a node, the mouse cursor is free to move within a predefined selection radius around it, symbolized by a gray circle. The link cursor shows the closest link, which will be selected upon passing the selection radius. (b) Link selection can also be performed at junctions of edge-bundles. Beyond a node or junction’s selection radius, the mouse cursor is constrained to slide along the selected link.

by links that can be long and cross many other links. Following a specific link can take a long time without zooming out, but zooming out makes it difficult to trace a link when other links overlap with it or cross it at shallow angles.

I had started developing an alternative navigation method to pan & zoom for RDF node-link diagrams in IsaViz while working on the Fresnel project (Section 2.2, Figure 2.7). A Fresnel lens, when hovering a resource node to which it was applicable, would temporarily bring inside the viewport the property values of that resource that were not currently visible but were declared by the lens as properties to display. The technique was using topological information about the RDF graph to facilitate resource-centric navigation in the data, a task that would otherwise have required the user to perform possibly tedious pan & zoom navigation actions depending on the graph’s size and layout.

In collaboration with members of the Aviz team at INRIA, we later pushed the concept of topology-aware navigation [MCH⁺09] (*Topology-Aware Navigation in Large Networks*, available in Appendix A). We designed two techniques, called Link Sliding and Bring & Go, and empirically compared them against pan & zoom techniques on representative graph navigation tasks. Both techniques are now available in ZGRViewer (Section 2.3.1) to navigate in arbitrary GraphViz layouts.

Following a route on a map, or a link in a graph visualization, is essentially a one-dimensional navigation task. However, traditional navigation techniques, such as pan & zoom, require the control of two or three degrees-of-freedom to accomplish the task effectively. The Link Sliding

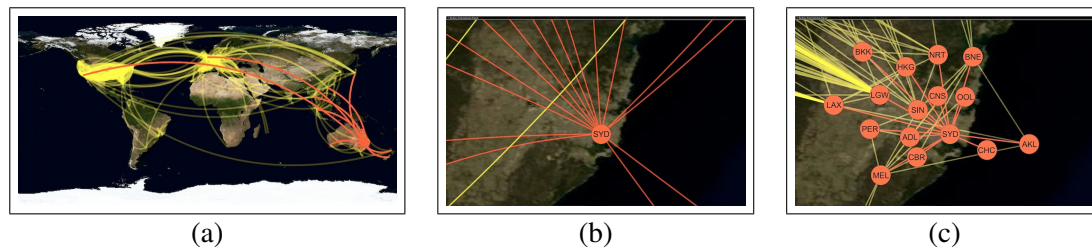


Figure 2.10 : (a) Highlighting all flights to/from Sydney, Australia. (b) Close-up on Sydney, with highlighting. (c) Bring and Go initiated on Sydney.

technique simplifies the control task by constraining motion to a single path (Figure 2.9). The user slides a *link cursor* along the link towards the destination node, as though sliding a bead on a wire. Changes in the direction of mouse movements are only necessary if the path curves sharply. Otherwise, the user may slide between two nodes by simply moving the mouse along the direction tangent to the path. This motion does not require a high degree of precision, as any movement perpendicular to the path is ignored, and motion stops at the destination node. The view is automatically panned to follow the mouse cursor, keeping it in its initial screen location, and the zoom level is adjusted so as to provide the user with additional context while sliding along the link. The technique also supports traversal of edge bundles, that are used to reduce visual clutter in dense graphs [Hol06]. Sliding along a bundle of links is identical to sliding along a single link until a junction in the bundle is reached. Around each bundle junction (Figure 2.9-b), a light-gray circle indicates a selection radius where the mouse cursor is detached from the link cursor, allowing the user to select an exiting link in the same manner as is done at the node where sliding got initiated. Users can jump between nearby links in a bundle by moving the mouse cursor rapidly away from the current link to which it is constrained. Isolated links strongly maintain the sliding constraint, as no other link can be reached within a single mouse motion event.

Link Sliding makes it easy to navigate along a given path. However, it does not help in the decision process that leads to the selection of one path among many potential candidates. This decision might depend on the type of arc to be followed when there are different types of paths. It might also depend on attributes of the node at the other end of the path. Having to navigate to the other end, in order to decide whether this is the path of interest or not, quickly becomes tedious as the number of connected arcs increases. The second technique we designed, called Bring & Go, is a generalization of the technique I had originally developed for Fresnel. Bring & Go is aimed at reducing the time and effort required to navigate to a distant location, as well as simplifying the selection of a destination. It accomplishes this by bringing adjacent nodes close to a node upon selection of the latter, and automatically transports the user to the selected point. To illustrate this, Figure 2.10-a shows a map of about 700 commercial flights connecting 232 airports. Highlighting (in red) gives a general idea of the number and location of airports connected to the currently selected node: Sydney International. At this scale, the node is difficult to select, being only 2-pixel large on a 24" display. Moreover, some parts

of the network are very crowded, making it difficult to visually follow the paths. One has to zoom-in to get detailed information such as airport names, thus losing context and moving all airports connected to Sydney out of the viewport (Figure 2.10-b). When selecting the node corresponding to Sydney, Bring & Go translates all airports connected to it inside the current viewport (Figure 2.10-c) using smooth animations to preserve perceptual continuity [RCM93]. The spline curves that represent links are smoothly flattened and brought inside the viewport, thus providing additional contextual information, such as the degree of connected nodes, that might help the user make her decision. For instance, she might be looking firstly for an airport hub, which would be more likely to offer her a direct flight to her final destination.

We conducted an experiment to compare the two techniques with simple visual augmentation of the methods currently available for navigating in node-link diagrams and maps: pan & zoom, with neighboring node highlighting, optionally augmented with an interactive bird's eye view. Participants were asked to perform various compound navigation tasks on an abstract graph, based on representative tasks from Lee *et al.*'s task taxonomy for graph visualization [LPP⁺06]: identifying all nodes connected to a given node, following a link, and returning to a previously visited link.

The results of this experiment clearly illustrated that using connectivity information as a basis for graph navigation can significantly improve task performance, while enhancing the quality of the user experience. Bring & Go was faster and easier to use than the others we tested. While the other techniques do use connectivity information to some extent, they rely primarily on the spatial layout of the graph, and on the motion of the user's view port in this space. The ability to see all connected nodes quickly, and traverse links rapidly, is probably what reduced participants' need to rely on their memory and eventually gave a clear advantage to Bring & Go. Link Sliding, on the contrary, did not fare as well as expected, performing the same as, or only slightly better than, pan & zoom augmented with a bird's eye view. Nevertheless, Link Sliding can be of interest for navigating networks such as route-maps, that are spatially embedded in a geographic context. Indeed, while Bring & Go works well for finding labeled nodes in abstract graphs, it provides very little information about the connected nodes' spatial context.

In the end, any system for visualizing large networks should benefit from some form of topology-aware navigation. As both the visual augmentation and the navigation techniques are triggered only in the context of links and nodes, they do not interfere with existing interaction techniques for spatial navigation. As each technique has its own unique strengths, a combination of two or more of them may be required for high-level navigation tasks.

Multi-scale Navigation in Large Datasets

The amount of data produced worldwide grows at an exponential rate. Twenty years ago, more new information had been produced in the previous thirty years than in the previous five thousands [Wur89]; and 90% of the data available today has been produced in the last two years [IBM12]. As mentioned earlier, those data come from everywhere: scientific research, georeferenced services, user-generated content and social networks, digital libraries, industrial processes and critical systems, finance and commerce. Telescopes can now produce extremely large images, such as Spitzer's 4.7 billion-pixel infra-red image of the inner part of our galaxy. Photographers can create huge images, such as the 26 gigapixel panorama of Paris based on 2,346 pictures stitched together, which was recently overtaken by an 80 gigapixel panorama of London. OpenStreetMap data, which range from an overview of the world down to detailed information at street level, fit in a 20-gigabyte compressed file (roughly 312GB uncompressed). Rasterizing the entire world at the highest level of detail would require an 18 peta ($18 \cdot 10^{15}$) pixel bitmap. The Google Maps/Google Earth dataset is likely much bigger than that.

Navigating in such large images and maps, or in visual representations of large networks and other structures generated via information visualization techniques [CMS99, War04], requires interaction techniques that scale both in terms of technical performance (graphical frame rate, information update rate) and user performance (providing effective task support, enhancing the user experience).

Joining the In-Situ project team at INRIA in late 2004 gave me the opportunity to focus my work on the design, implementation and evaluation of interaction techniques for navigating large information spaces. The following sections summarize the results obtained so far, organized in two main themes: the design and evaluation of novel interaction techniques for multi-scale navigation, including techniques for interacting with wall-sized displays; and the engineering of multi-scale interactive systems, focusing on how to make the programming of advanced visual interfaces easier for software developers.

3.1 Design and Evaluation of Interaction Techniques

3.1.1 An Operationalization of Multi-scale Search

Zoomable User Interfaces (ZUI) [PF93] and multi-scale interfaces at large have generated a growing interest over the past two decades as a powerful way of representing, navigating and manipulating large sets of data. A number of techniques have been designed and implemented, that can be categorized in three primary interface schemes [CKB08]: zooming, overview+detail, and focus+context. Up until recently, the efficiency of these techniques had been evaluated with two kinds of experimental studies: usability studies based on domain-specific tasks, and controlled experiments based on multi-scale versions of Fitts' pointing paradigm [GB04].

The usability studies that relied on domain-specific tasks such as searching for items on geographical maps [HBP02], comparing hierarchical data structures [NBM⁺06], or reading

textual documents [HF01] had typically produced inconclusive and sometimes contradictory results. More precisely, experimental findings varied from study to study, but since application domains varied dramatically, these findings could neither be compared nor generalized. Such results had to be expected since the performance of a given technique is indeed dependent on its context of use [ABM05].

The goal of this work [PAB07] (*Pointing and Beyond: an Operationalization and Preliminary Evaluation of Multi-scale Searching*, available in Appendix A), conducted together with two members of the team, Caroline Appert and Michel Beaudouin-Lafon, was to get a better understanding of the fundamental aspects of multi-scale navigation. A better understanding could help explain – or even predict – experimental results, therefore saving valuable time and allowing better exploration of novel techniques. Fitts’ pointing paradigm provides such a fundamental tool for exploring and understanding the elementary task of reaching a known target as fast as possible. Originally devised to study pointing in the real world [Fit54], it has been used repeatedly in HCI for evaluating a variety of pointing techniques and devices. Fitts’ law has proven remarkably robust, to the point of being used as part of an ISO standard for pointing devices [SM04]. Fitts’ pointing task has also been used with multi-scale interfaces and it has been shown that Fitts’ law still applies for pointing targets with pan & zoom [GB04]. In particular, it has been shown that Fitts’ paradigm could address navigation, not just pointing, in interfaces that require scrolling or zooming.

While Fitts’ pointing paradigm is very powerful, it models a very specific task: that of reaching a target whose location is known to the user. However, this scenario only captures one of several navigation tasks in multi-scale worlds. Users might only have partial information about the target’s location and appearance, thus requiring them to search for potential targets and get more details about each one until the actual target is identified. Consider for example a user searching for a salt lake with particular visual characteristics on a world map, only knowing that it is in the Andes. The strategy first consists in zooming towards South America to then inspect each potential target one by one, zooming in to discover that it is not the right one, zooming out, maybe as far as the whole cordillera, and zooming in to the next potential target until the right *salar* is found. Exploring large spaces in search of a particular target differs from pure pointing, as it requires users to perform additional motor actions to identify the target. In the same way as Fitts’ reciprocal pointing task operationalized the task of reaching a known target, we proposed to operationalize the above search task in a way that was easily amenable to controlled experiments.

Our operationalization of multi-scale search is based on an abstract search task that consists in finding a target among a set of objects as quick as possible while minimizing the number of errors. To find the target, users have to navigate in both space and scale to a position that reveals enough details about each object, in order to decide whether it is the target or a distractor. Initially, users make a blind choice of a potential target at a high scale and navigate to it to acquire enough information. If it is a distractor, they have to navigate to another object, typically by zooming-out, panning, then zooming-in.

One of the main contributions of this operationalization, that was later reused in other experiments conducted in the team and elsewhere [JGE12], is the manner in which we minimized the influence of luck when looking for one particular object in a set of distractors. The experimental setup consists of a multi-scale world containing a set of n objects, one of them being the target and the others distractors. We defined the “quantity” of exploration as the number k of distractors that users have to visit before finding the target: the larger the number of visited distractors, the larger the quantity of exploration. k is probabilistically dependent on n : the larger the number of objects, the higher the probability of having a large number of objects to visit before reaching the target. We controlled this parameter by forcing participants to visit a predefined number of objects before finding the target. The “trick” lied in how we were choosing the target object. If we had chosen *a priori* which object was the target, participants could have found it immediately by chance, or on the contrary they could have spent a lot of time searching for it. This uncontrolled factor would have had a significant impact on our measurements. We thus designed our experiment to ensure that the target was the k^{th} object visited, no matter the order of exploration chosen by each participant. In other words, what object in the whole set would be the target was decided dynamically, at the last moment, depending on the value of k for the current trial and on how many objects the participant had already inspected.

We ran a first experiment based on this operationalization, whose purpose was to evaluate how four multi-scale navigation techniques perform in one particular configuration of a multi-scale world: classical pan & zoom, overview + detail as typically found in mapping applications, and two focus + context techniques: a constrained fisheye lens [CM01] and a variation on the DragMag image magnifier [WL95]. The results of this experiment indicated that, in this context, pan & zoom combined with an overview is the most efficient technique of all four, and that focus + context techniques perform better than pan & zoom alone. The results are valid for this type of configuration only, and additional experiments should be conducted to find out whether they also apply to other configurations of multi-scale worlds.

3.1.2 A Design Space for Focus+Context Interaction Techniques

The various types of interface schemes that we tested in the experiment mentioned above are actually complementary. Pan & zoom navigation is often augmented with an interactive bird’s eye view (overview + detail). Magnification lenses [CM01] or widgets such as the Drag-Mag [WL95] (both focus + context) can be combined with classical pan & zoom, in elaborate techniques that enable users to probe regions of the space currently observed in the main viewport [PAB07]. Overview + detail techniques [PCS95] usually put the context view in a small inset located in a corner of the screen, leaving most of the latter to the detailed view, while focus + context techniques do the opposite: the context occupies the whole screen except for a small area that provides an in-place, smoothly integrated, magnification of a limited region of the context [CKB08]. While overview + detail techniques are generally favored and have been shown to perform well in some situations [HBP02, NS00, PAB07], there are cases where they show their limits: for instance, when navigating a map of a densely populated region to look for particular localities, overview + detail techniques can only use a few pixels to display each of them in the context view. On the contrary, focus + context techniques can convey additional

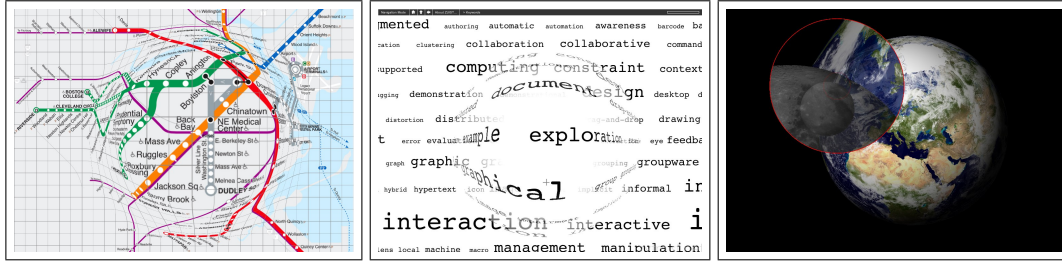


Figure 3.1 : Three different lenses obtained with minor modifications to the scale and compositing functions: (left) a distortion lens on a high-resolution bitmap (subway map), (middle) a hovering lens on 2D vector graphics and text (keyword tag cloud), (right) a speed-coupled blending lens on a 3D model of the Moon orbiting the Earth.

information in the context view, such as the localities' names, thus providing users with more contextual information that can guide navigation. This happens, however, at the expense of the focus region's size.

Focus + context techniques have been studied for some time, but their adoption remains limited. This is likely due to comprehension and low-level interaction problems related to how the transition between the context and the magnified focus region is achieved [CCF97, Gut02]. Many of the transitions described in the literature are inspired by the physical world and are presented through metaphors such as magnifying glasses, rubber sheets [SSTR93], and more generally surface deformations [CCF95]; in other words, spatial transitions that cause various problems likely to hinder performance. For instance, simple magnifying glasses create occlusion of the immediate context adjacent to the magnified region [RCM93]; graphical fisheyes [SB94], also known as distortion lenses, make it difficult to acquire targets [Gut02], especially for high magnification factors.

In collaboration with Caroline Appert and Olivier Bau, who were then both PhD students at Université Paris-Sud, we worked on the definition of the Sigma Lens framework. The foundational idea of Sigma lenses was that other dimensions than spatial distortion, readily available in the electronic world, could be used to provide more efficient transitions between the focus and context regions of a bi-focal visualization based on constrained lenses.

The framework is based on the general observation that no matter the representation and underlying graphics API, the process of rendering focus+context magnifications consists in rendering a subregion F of the current representation C at a larger scale and with more detail, and integrating F into C through a non-linear transformation to achieve a smooth transition between the two. Sigma Lenses extend this general process by defining a design space of transitions between focus and context. Transitions are combinations of dynamic displacement and compositing functions, that make it possible to create a variety of lenses that use techniques other than spatial distortion to achieve smooth transitions between focus and context, and whose properties adapt to the users' actions, all in an effort to facilitate interaction.

Definition 1: displacement and compositing function $\mathcal{R}$

$$\mathcal{R}(x, y) = \begin{cases} (x_c + \frac{x-x_c}{MM}, y_c + \frac{y-y_c}{MM}) \otimes_{\alpha_{FT}} (x, y) & \{\forall(x, y) | \mathcal{D}(x, y) \leq R_I\} \quad (1.1) \\ (x_c + \frac{x-x_c}{\mathcal{G}_{scale}(\mathcal{D}(x, y))}, y_c + \frac{y-y_c}{\mathcal{G}_{scale}(\mathcal{D}(x, y))}) \otimes_{\mathcal{G}_{comp}(\mathcal{D}(x, y))} (x, y) & \{\forall(x, y) | R_I < \mathcal{D}(x, y) < R_O\} \quad (1.2) \\ (x, y) & \{\forall(x, y) | \mathcal{D}(x, y) \geq R_O\} \quad (1.3) \end{cases}$$

We worked on both the design and evaluation of novel techniques in this design space [PA08], and on a general, representation-independent approach to the framework's implementation, that is based on a rendering technique founded on a unified model that can be integrated with minimal effort in different graphics frameworks, ranging from 3D graphics consisting of complex textured meshes to rich multi-scale 2D graphics combining text, bitmaps and vector graphics [PBA10] (*Representation-Independent In-Place Magnification with Sigma Lenses*, available in Appendix A), as illustrated in Figure 3.1.

Our general approach is positioned at a level of abstraction high enough for the model to be applicable to a variety of graphics frameworks, requiring the underlying libraries to provide as small a number of features as possible. The underlying graphics library must allow i) for the scene to be rendered at different levels of detail, and ii) for the pixels that constitute the two rendered images (the context region and the lens region) to be manipulated and composited before the actual rendering to the screen occurs. We developed two different implementations, one in Java for multi-scale 2D graphics that is now part of the core module of the ZVTM zoomable user interface toolkit (Section 3.2.1), the other for OpenGL 3D scenes, that takes advantage of programmable graphics hardware (lenses are written with GLSL, the OpenGL Shading Language). The approach basically consists in transforming the representation at the pixel level after it has been rendered, independently of how it was rendered. Our technique consists in asking the underlying graphics library for two separate rendering passes. One corresponds to what is seen in the context region and is stored in the context buffer, whose dimensions match that of the final viewing window displayed to the user. The other rendering pass corresponds to what is seen in the lens region. The final viewing window displayed on screen can then be obtained through the arbitrary transformation and composition of pixels from both buffers. This includes displacement and compositing functions that will control the transition between the focus and context regions, summarized in Definition 1. The flat-top region corresponds to case (1.1), the transition to case (1.2), and the region beyond the lens boundaries, i.e., the context, corresponds to case (1.3).

The standard transformation performed by graphical fisheyes consists in displacing all points in the focus buffer to achieve a smooth transition between focus and context through spatial distortion (Figure 3.2-a). This type of transformation can be defined through a drop-off function which models the magnification profile of the lens. The drop-off function is defined as:

$$\mathcal{G}_{scale} : d \mapsto s$$

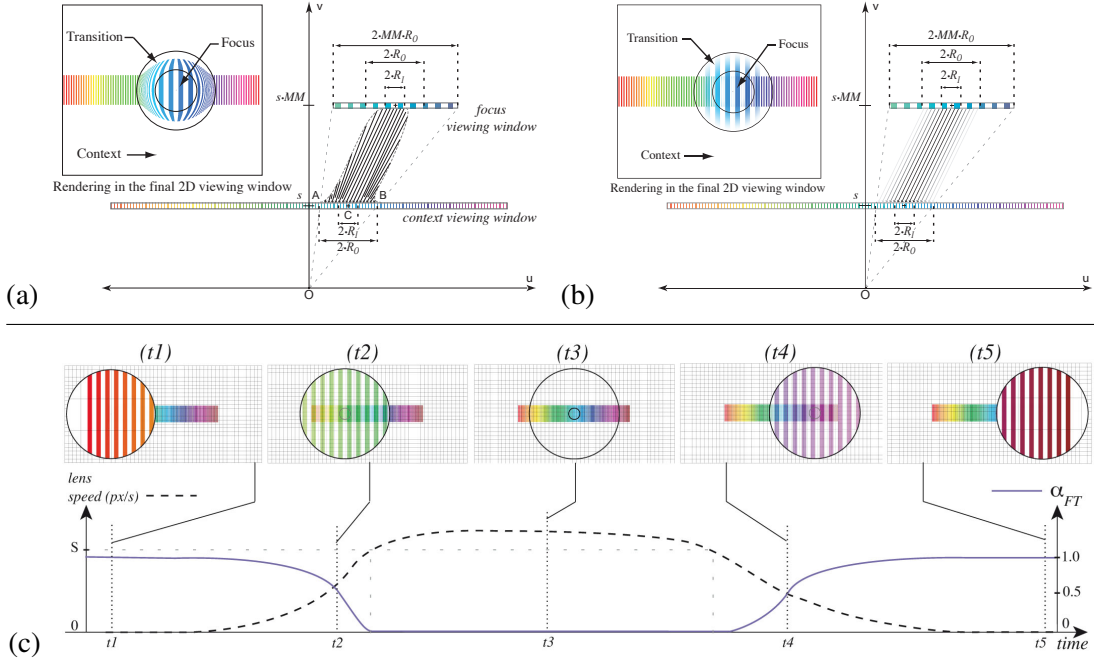


Figure 3.2 : Sigma lenses in a 1+1D space-scale diagram, and corresponding 2D rendering: (a) Gaussian distortion lens, (b) Blending lens. Speed-Coupled Blending Lens moving from left to right over time (c).

where d is the distance from the center of the lens and s is a scaling factor. Distance d is obtained from an arbitrary distance function $\mathcal{D}$. A Gaussian-like profile is often used to define drop-off function $\mathcal{G}_{scale}$, as it provides one of the smoothest visual transitions between focus and context. Distance functions producing basic regular lens shapes are easily obtained through $L(P)$ -metrics [CM01]:

$$\mathcal{D} : (x, y) \mapsto \sqrt[P]{|x - x_c|^P + |y - y_c|^P}$$

where (x, y) are the coordinates of a point seen through a lens centered in (x_c, y_c) , and $P \in \mathbb{N}^*$. $P = 2$ corresponds to a circular lens and $P = \infty$ to a square lens.

In our approach, the overall process consists in applying a displacement function to all pixels in the lens buffer that fall into the transition zone. Pixels of the lens buffer can then be composited with those of the context buffer that fall into the lens region. When only interested in spatial distortion, generating the final representation simply consists in replacing pixels in the lens region of the context buffer by those of the lens buffer; in other words compositing them with the **over** operator ($\alpha = 1.0$). But other values of α and other operators in Porter & Duff's rich algebra [PD84] can be used to achieve interesting visual effects. It is for instance possible to obtain smooth, distortion-free transitions between focus and context by applying an alpha blending gradient centered on the lens (Figure 3.2-b).

The rendering of a point (x, y) in the final viewing window is controlled by function $\mathcal{R}$ (see Definition 1), where

$$p_{lens} \otimes_{\alpha} p_{context}$$

denotes the pixel resulting from alpha blending a pixel from the lens buffer and another from the context buffer with an alpha value of α . As with scale for distortion lenses, the alpha blending gradient can be defined by a drop-off function that maps a translucence level to a point (x, y) located at a distance d from the lens center:

$$\mathcal{G}_{comp} : d \mapsto \alpha$$

where α is an alpha blending value in $[0, \alpha_{FT}]$, α_{FT} being the translucence level used in the flat-top of the lens.

The Sigma Lens framework also allows for lens properties such as magnification factor, radius or flat-top opacity to vary over time. The first example of lens to make use of dynamic properties was Gutwin's speed-coupled flattening lens [Gut02], which uses the lens' dynamics (velocity and acceleration) to automatically control magnification (Figure 3.4-a). By canceling distortion during focus targeting, speed-coupled flattening lenses improve the usability of distortion lenses. Basically, magnification decreases toward 1.0 as the speed of the lens (operated by the user) increases, therefore flattening the lens into the context, and increases back to its original value as the lens comes to a full stop. Such behavior can easily be implemented in our approach using a simple interpolated low-pass filter (see [PA08] for detailed information). Other properties can be made speed-dependent, including the radii, as well as the translucence value in the lens' flat-top α_{FT} . For instance, the speed-coupled blending lens (Figure 3.2-c and Figure 3.4-b) is also easily modeled. This lens features a larger flat-top area compared to lenses of the same size that feature a transition zone. This makes focus targeting tasks easier for the user from a purely motor perspective, but the occlusion stemming from the absence of a smooth transition zone counterbalances this theoretical advantage. The occlusion problem is addressed by coupling α_{FT} to the speed of the lens: the lens becomes increasingly translucent as it is moved faster, and conversely.

Various Sigma lens designs, including the speed-coupled blending lens described above, were evaluated to assess the impact of the various transition types on users' focus targeting performance. This task consists in putting a given target in the flat-top of the lens and is one of the building blocks of many higher-level navigation tasks such as multi-scale searching (Section 3.1.1). A total of four experiments were run in different contexts, ranging from very abstract and basic environments as those generally used in pointing experiments [SM04] to networks represented as vector graphics, or labels overlaid on complex satellite imagery with a varying level of visibility. The overall results of those experiments were that one of the new designs enabled by the Sigma lens framework, namely the speed-coupled blending lens, outperformed all others on this task, in all contexts we tested.

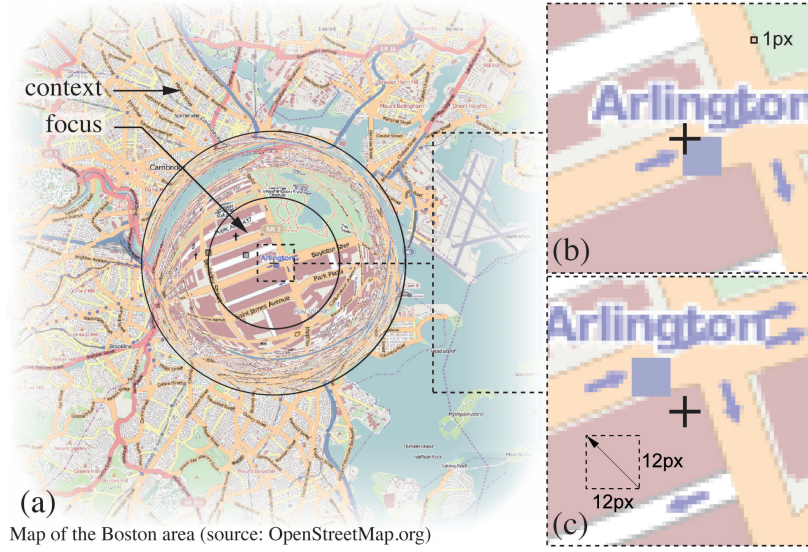


Figure 3.3 : (a) In-place magnification by a factor of 12 of the south-west corner of the Boston Public Garden. Between (b) and (c), the lens has moved by 1 unit of the input device, corresponding to 1 pixel in the context, but the magnified region is offset by 12 pixels. Objects can thus be difficult or even impossible to select; even if their visual size is above what is usually considered a small target (less than 5 pixels). The square representing Arlington station is 9-pixel wide, yet its motor size is only 1 pixel.

One problem related to focus targeting that was not addressed in the original work on Sigma lenses is that of the mismatch between visual and motor precision in the magnified region. Early implementations of magnification techniques only magnified the pixels of the context by duplicating them without adding more detail, thus severely limiting the range of useful magnification factors (up to 4x). Newer implementations, including Sigma lenses, do provide more detail as magnification increases. Theoretically, this means that any magnification factor can be applied, if relevant data is available. In practice, this is not the case as another problem arises that gets worse as magnification increases: *quantization*.

Lenses are most often coupled with the cursor and centered on it. The cursor, and thus the lens, are operated at context scale. This allows for fast repositioning of the lens in the information space, since moving the input device by one unit makes the lens move by one pixel at context scale. However, this also means that when moving the input device by one unit, the representation in the magnified region is offset by MM pixels, where MM is the focus' magnification factor. This means that only one pixel every MM pixels can fall below the cursor in the magnified region. In other words some pixels are unreachable, as illustrated in Figure 3.3.

This quantization problem has been a strong limiting factor in terms of the range of magnification factors that can be used in practice; the upper limit reported in the literature rarely

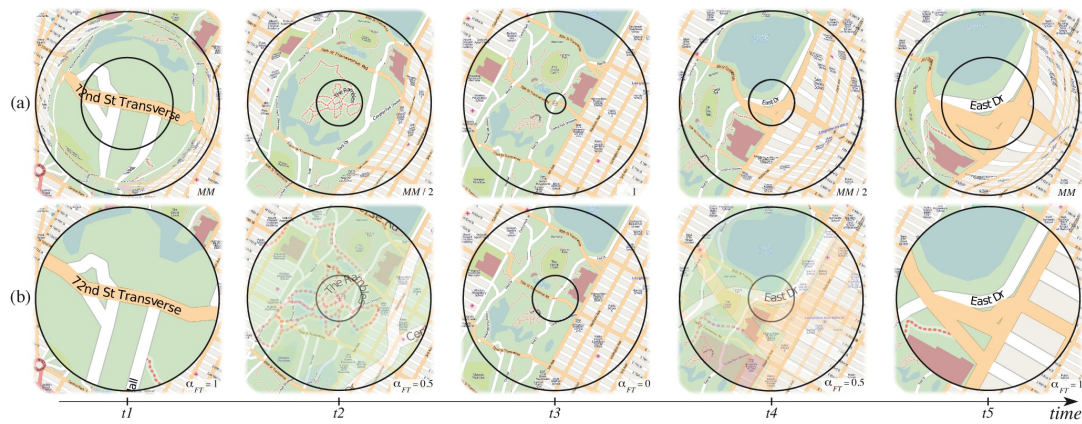


Figure 3.4 : Behavior of two Sigma lenses during a focus targeting task ending on *East Drive* in Central Park. (a) As speed increases, the *speed-coupled flattening lens* smoothly flattens itself into the context (from $t1$ to $t3$), and gradually reverts to its original magnification factor when the target has been reached ($t4$ and $t5$). The inner circle delimits the region magnified in the flat-top. (b) As speed increases, the *speed-coupled blending lens* smoothly fades into the context (from $t1$ to $t3$), and gradually fades back in when the target has been reached ($t4$ and $t5$). The inner circle fades in as the lens fades out; it delimits which region of the context gets magnified in the lens. The magnification factor remains constant.

exceeds 8x, a value relatively low compared to the ranges of scale encountered in the information spaces mentioned throughout this memoir. Together with Caroline Appert and Olivier Chapuis, we designed and evaluated three novel interaction techniques that enable fast navigation and high-precision focus-targeting and object selection in the magnified region [ACP10] (*High-precision Magnification Lenses*, available in Appendix A). The three techniques use different strategies: one continuously adapts motor precision to navigation speed, while the two others use a discrete switch between two levels of precision (focus and context): one using an additional input channel, e.g., a modifier key such as shift; the latter by decoupling the cursor from the lens' center. Those three techniques were integrate in the speed-dependent visual behaviors from the Sigma Lens framework, with the resulting hybrid lenses significantly enhancing focus targeting performance and allowing for higher magnification factors, typically up to 12x, as shown in a series of experiments reported in [ACP10].

Work on focus+context visualization techniques is now continuing mostly through Cyprien Pindat's PhD (co-advised with Claude Puech) who got interested in the concept of *adaptive lenses* [PPCP12] that dynamically adapt their shape to provide more relevant magnifications than the regular, statically defined lenses that currently represent the state-of-the-art.

As can be seen in the various publications about Sigma lenses and high-precision lenses, the operationalization of the focus targeting task that served as a basis for the evaluation of our new lens designs is strongly inspired by the framework provided by Fitts' law [Fit54,

SM04]. The study of pointing, and the design and evaluation of pointing facilitation techniques is also a research theme developed in the team, and I had the opportunity to collaborate with team members, first and foremost with Olivier Chapuis, on this topic. This led for instance to the design of DynaSpot [CLP09] (*DynaSpot: Speed-dependent Area Cursor* , available in Appendix A), a new pointing facilitation technique for acquiring targets based on the area cursor [KB95]. DynaSpot is further related to the work on Sigma lenses as it also features a speed-dependent behavior: the technique couples the cursor's activation area with its speed, making it grow as a function of speed up to a maximum size, typically set to a few dozen pixels, and behaving like a point cursor at low speed or when motionless, thus allowing users to access all conventional point cursor interactions seamlessly, including empty space and region selections without the need for an explicit mode switch.

3.1.3 Focus+Context Interaction for Time-series

Beyond the multiple types of datasets amenable to multi-scale two-dimensional representations that were discussed so far (trees, networks and other types of node-link diagrams, maps, collections of documents, etc.), another type of data are often produced in massive quantities and call for multi-scale visualization techniques: time-series. Visual representations of time-series make it significantly easier for users to discover trends and patterns at different scales, but also to identify anomalies in the data [BAP⁺05, MMKN08]. However, basic time-series visualizations using line plots do not scale well; and as time-series data are often very large, featuring multiple, possibly heterogeneous, dependent variables measured for long periods of time and/or at high sampling rates, visualization of real-world time-series data poses significant challenges and has been an active area of research for many years.

While many interactive visualization tools have been developed to address this scalability problem, offering innovative alternatives to the common line plot visualizations or enhancing the visualization with advanced interactive features, there has been comparatively little research on how to support the more elaborate tasks typically associated with the exploratory visual analysis of time-series, e.g., visualizing derived values, identifying correlations, or identifying anomalies beyond obvious outliers. Such tasks typically require deriving new time-series from the data, visualizing those time-series and relating them to the original data plots.

Visual exploration techniques take advantage of human abilities to drive the data exploration process and are especially useful for undirected searches, when users know little about the data or have only vague exploration goals [Kei02, Shn96]. As emphasized by Keim's visual analytics mantra – “*Analyze first, Show the important, Zoom, Filter and analyze further, Details on demand*” [KMSZ06], that draws on Shneiderman's visual information seeking mantra [Shn96] – this process is iterative. For it to be efficient, visual representations, that support human judgment, and interactions, that re-parameterize the visual representation, should be tightly integrated, enabling users to quickly choose and refine parameter values that best suit the task at hand [AMM⁺08]. New plots derived from the original data should be put in context and made easy to relate both to the original data and to other plots that have been derived as part of the exploratory process.

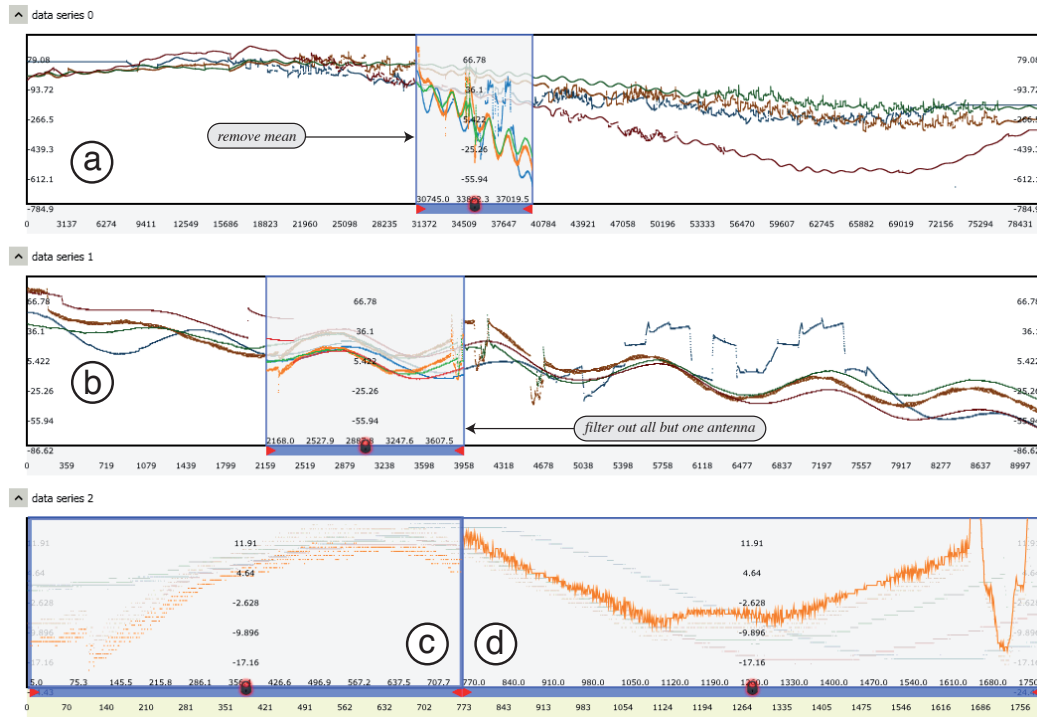


Figure 3.5 : Exploring ALMA Line Length Correction Stretcher Voltage plots: (a) two-day overview at a sampling rate of one second; (b) magnification of the 5 hours seen through the *remove mean* lens applied in (a); magnification of the 50 minutes for a single antenna seen through a filtering lens, rendered in scatterplot mode (c) and line-plot mode (d).

Together with members of the Dynamic Graphics Project at the University of Toronto¹, we recently investigated the applicability of lens-based visualizations to the exploratory visual analysis of large time-series data [ZCPB11] (*Exploratory Analysis of Time-series with ChronoLenses*, available in Appendix A). We designed ChronoLenses, illustrated in Figure 3.5.

ChronoLenses, as many other types of lenses, delimit a region of interest in the data to be put in focus. But as opposed to Sigma lenses and related detail-in-context techniques, the visual transformation is not limited to magnification, but also includes visual filtering [Fur86] and other arbitrary graphical transformations of the underlying content [BSP⁺93]. Based on the metaphor of direct manipulation [Shn83], ChronoLenses perform on-the-fly transformation of the data points in their focus area, tightly integrating visual analysis with interaction. Users can build pipelines composed of lenses performing various transformations on the data (e.g., remove mean, compute 1st derivative, auto-correlation), effectively creating flexible and reusable time-series visual analysis interfaces. At any moment, users can change the parameters of already created lenses, with the modifications instantaneously propagating down through the pipeline, providing immediate visual feedback that supports the iterative exploration process.

¹<http://www.dgp.toronto.edu>



© CNRS Photothèque / Cyril Frésillon

Figure 3.6 : Left: OpenStreetMap of Manhattan. The inset illustrates the amount of information available in a $9\text{cm} \times 5\text{cm}$ area. Right: zoomed-in view of the inner part of our galaxy in the infrared domain ($396\,032 \times 27\,040$ pixels), taken by the Spitzer telescope.

The design of ChronoLenses was informed by a set of design requirements. Those requirements were derived from the low-level tasks identified through interviews with expert users from varied domains including operations monitoring and control for the ALMA radio-observatory, environmental research related to weather forecast, as well as financial and network streaming data analysis. Figure 3.5 shows an example of multi-scale visualization of monitoring data coming from the ALMA radio-telescope (Section 3.2.1), where time-series visualization is used by both operators and astronomers for a variety of tasks, ranging from checking some of the thousands of monitor points in the system to performing scientific data quality assurance during observations.

3.1.4 Interacting with Wall-sized Displays

Problems of multi-scale visualization are not limited to desktop/workstation environments. They also arise on small devices such as smartphones and tablets, as well as on large screens and wall-sized displays. The latter, however, also offer tremendous new opportunities for multi-scale visualization, when *physically large* entails *significantly higher number of pixels*.

Many of the early so-called *large displays* or even *wall-sized displays* were large in terms of physical size indeed, but did not feature a number of pixel that was dramatically different from desktop setups. In the last five years, wall-sized displays have evolved from arrays of tiled projectors to setups that consist of juxtaposed LCD panels. The latter are often called *ultra-high-resolution* displays to emphasize their significantly higher display capacity compared to projector-based *very-high-resolution* displays. For instance, the display depicted in Figure 3.6 features a resolution of $20\,480 \times 6\,400 \simeq 131$ megapixels on a $5.5\text{m} \times 1.8\text{m}$ surface ($\simeq 100\text{ppi}$ resolution). These displays typically accommodate several hundred megapixels. They enable the visualization of truly massive datasets, and afford a more physical, as opposed to virtual, way of navigating in the data [BNB07] than desktop displays or even large projection-based displays. They can represent the data with a high level of detail while retain-

ing context [NWP⁺11], and enable the juxtaposition of data in various forms. Applications include scientific visualization, automotive or airplane design, network monitoring, geospatial intelligence, crisis management centers and control rooms.

From 2008 to 2012, I coordinated the WILD project² (Wall-sized Interaction with Large Datasets [BCE⁺12]). WILD is an experimental, ultra-high-resolution interactive platform for conducting research on collaborative human-computer interaction and the visualization of massive datasets. While other ultra-high-resolution wall-sized displays had been built before in other laboratories, most of these were focusing on the technical aspects of how to operate such platforms: how to display complex graphics, how to stream data; but they did not pay much attention to issues related to interaction with such displays, and offered only poor interaction capabilities such as wireless mouse & keyboard on a stand or in some cases gyroscopic mice. One exception was the setup built by the DGP lab at the University of Toronto, but this wall-display was projection-based, and featured a resolution much lower than what could now be achieved using juxtaposed LCD tiles. We were interested in designing and developing novel interaction and visualization techniques for massive scientific datasets³ that would enable users to visualization and manipulate data directly and would foster physical navigation patterns afforded by the significantly increased display capacity of ultra-high-resolution walls.

The platform, which now hosts many projects and has become one of the nodes of the larger Digiscope project⁴, consists of a wall-sized display, a 3D motion capture system, a tabletop interactive surface and mobile devices such as smartphones and tablets. Research on WILD is organized around three main themes:

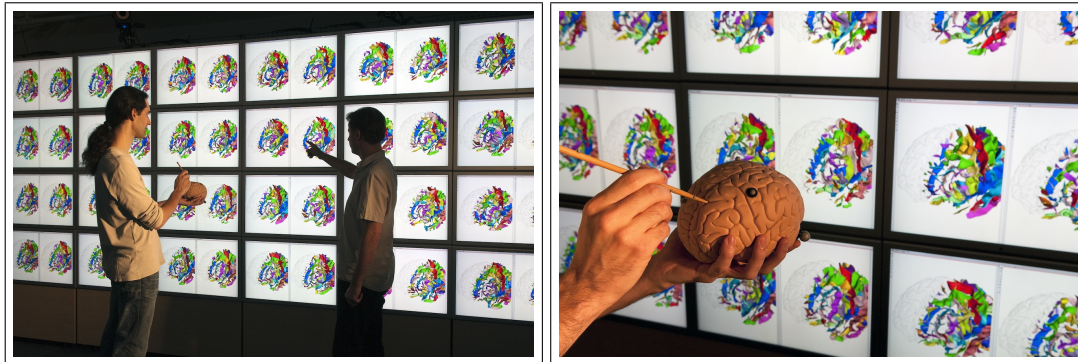
- Multi-scale interaction: as mentioned earlier, the ultra-high resolution affords multi-scale interaction through simple physical navigation: approaching the wall reveals details⁵, stepping back gives an overview of the information space. Visualizations are not necessarily made of one single image or 3D model rendered at a very high resolution; they can be made of multiple coordinated views of instances of the same type, such as the synchronized brain scans in Figure 3.7, or even heterogeneous data related to the same analysis task (PDF documents, images, interactive 3D graphics, maps, charts, spreadsheets, etc.). We are interested in designing interaction and visualization techniques that take advantage of these properties.
- Multi-user and multi-surface interaction: ultra-high-resolution wall-sized displays are well-suited to collaborative analysis tasks. Users can interact simultaneously on the same

²Collaborative project involving three teams: In-Situ (*situated interaction* - INRIA, CNRS & Université Paris-Sud), Aviz (visual analytics - INRIA) and AMI (architectures and models for interaction - CNRS & Université Paris-Sud). Coordinators and Principal Investigators: Emmanuel Pietriga and Michel Beaudouin-Lafon. More information at <http://insitu.lri.fr/Projects/WILD>

³The project involved associate laboratories from other scientific disciplines that were interested in using the platform for the visual collaborative analysis of their data, including astrophysics with the Institut d'Astrophysique Spatiale (IAS), particle physics with the Laboratoire de l'Accélérateur Linéaire (LAL) and neuroanatomy with the Laboratoire de Neuroimagerie Assistée par Ordinateur (LNAO/Neurospin).

⁴<http://www.digiscope.fr>

⁵At $\simeq 100\text{ppi}$, text elements rendered with fonts as small as 8pt are still perfectly legible.



© CNRS Photothèque / Cyril Frésillon

Figure 3.7 : Manipulating 64 synchronized 3D brain scans using physical props.

dataset and exchange data through the wall, that can be used as a shared information space. We are interested in collaborative interactions involving multiple display and input surfaces, including smartphones, tablets, and tabletop surfaces connected to the wall display, and the exchange of data across devices.

- Facilitating software development for such platforms: This particular research theme is developed further in Section 3.2.2. Ultra-high-resolution wall displays are generally driven by clusters of computers, making the development of visualizations challenging as the data and/or rendering pipeline has to be distributed across multiple computers to eventually form a coherent result. Input devices also have to work seamlessly across surfaces, and across the multiple computers that drive a single large surface such as the wall display. In addition, conventional input devices such as mouse and keyboard are not adapted to environments like WILD. They significantly impede physical navigation and are not optimal for cooperative work. Mobile devices, motion capture systems and other heterogeneous devices are better enabling technologies for the design of interaction techniques adapted to this context. However, they make software development more complex.

The following sections give an overview of the research projects related to the design and evaluation of interaction techniques that I was directly involved in. Several of these research projects, and the WILD project in general, have been informed by participatory design sessions involving end users, mostly astrophysicists and experts in neuroimaging: workshops were held to identify their needs, create early prototypes, collect their ideas for improvements and refine the prototypes, all based on real usage scenarios [BCE⁺12].

Distant Ultra-High-Precision Pointing

One of the first problem we investigated was that of high-precision distant (or remote) pointing, as part of Mathieu Nancel's PhD, co-advised with Michel Beaudouin-Lafon. Distant pointing at large displays had been studied in various contexts, ranging from low resolution displays

to high-resolution back-projected walls. However, it had not been studied in the context of ultra-high resolution walls that can display much smaller visual elements that users must still be able to select, as illustrated in Figure 3.6-left.

At such high resolution, basic modeless techniques do not work. The well-known ray-casting technique, also called laser pointing [ON01], is not precise enough. The technique extends the user's arm or a hand-held device with an imaginary ray whose intersection with the wall display is highlighted. Ray casting degrades quickly with distance to the wall, because hand tremor and involuntary motion due to fatigue are amplified as the user is farther away from the display surface [MBN⁺02, OS02]. Relative techniques [FVB06, MJ01] achieve better precision but do not scale to large surfaces because of the need for clutching [CVBC08]. Some techniques combine absolute and relative pointing [MI09, VB05], but they have been designed and evaluated on displays that were either significantly smaller, or of lower resolution, or both. It was thus unclear how they would fare in the context of ultra-high-resolution, very large displays such as WILD. In this context, our question was: given the very high pixel density, is there a pointing technique that enables efficient selection of both large and small targets at a distance, and if not, can we design one?

We investigated this question by first identifying the limits of modeless techniques in a formative user study. We then considered techniques that feature two levels of precision, a coarse positioning mode to approach the area of the target and a precise pointing mode for acquiring the target, with a method to calibrate the parameters of those two modes. We introduced new techniques and compared them to adaptations of existing ones [FKK07, VB05], and found that techniques combining ray casting for coarse pointing and relative position or angular movements for precise adjustments of the cursor's position enable the selection of targets as small as 4 millimeters while standing 2 meters away from the display⁶. See [NPBL11] for more detail.

Multi-scale Navigation on Wall Displays

Another problem we investigated as part of Mathieu Nancel's PhD is that of how to perform pan & zoom navigation in mid air while standing at arbitrary locations in front of wall-sized displays [NWP⁺11] (*Mid-air Pan-and-Zoom on Wall-sized Displays* , available in Appendix A).

While ultra-high-resolution displays typically feature 40 to 100 times more pixels than a conventional workstation's screen(s), datasets increase in size faster than displays increase in dimensions and pixel density. On one side, WILD consists of thirty-two 30-inch tiled monitors and can display a "mere" 131 million pixels; NASA's Hyperwall-2 and the HyperWall at UCSD, to our knowledge the largest walls built to date, only double that number, and do so by adding some screens that users cannot reach. On the other side, as we observed earlier, datasets in many domains dwarf these display capacities: Spitzer's image of the inner part of the galaxy

⁶In comparison, the smallest target sizes reported in earlier studies on wall displays ranged from 9 centimeters [FKGR09] to 1.6 centimeters [VB05].

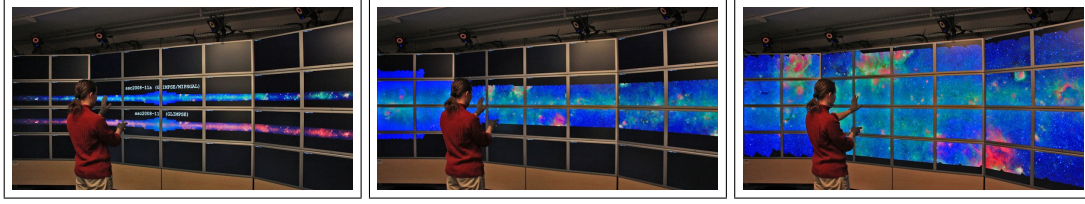


Figure 3.8 : Panning and zooming in Spitzer’s $396\,032 \times 27\,040 = 4.7$ billion pixels images of the inner part of our galaxy.

contains 4.7 billion pixels; one can find 26 to 80 gigapixel panoramas of cities on the Web; rasterizing OpenStreetMap data at street level would require an 18 petapixel bitmap. Virtual navigation is thus required even on wall-sized displays, as datasets can be several orders of magnitude too large to fit on them (Figure 3.8).

As discussed in Section 3.1, many interaction techniques have been specifically designed to help users navigate large multiscale worlds on desktop computers, using zooming and associated interface schemes [CKB08]. However, high-resolution wall-sized displays pose different sets of trade-offs. It is critical to their success that interaction techniques account for both the physical characteristics of the environment and the context of use, including cooperative work aspects. Input should be location-independent and should require neither a hard surface such as a desk nor clumsy equipment: users should have the ability to move freely in front of the display and interact at a distance [BNB07, YHN07], as we discussed earlier. This precludes use of conventional input devices such as keyboards and mice, as well as newer interaction techniques that do not meet requirements such as being location-independent [MRB05, MLG10].

Our goal was to study different families of location-independent, mid-air input techniques for pan & zoom navigation on wall-sized displays. More specifically, we were seeking to answer questions related to the performance and subjective preferences of users, including: Beyond their almost universal appeal, do gestures performed in free space work better than those input via devices operated in mid-air? Is bimanual interaction more efficient in this context? Is it more tiring? Do circular, continuous gestures perform better than those that require clutching (restoring the hand or finger to a more comfortable posture)? We grounded our work on both theoretical and experimental work on bimanual input [BM86, Gui87, LZB98], the influence of limb segments on input performance [BM97, ZMB96], on types of gestures [MH04b, Whe03] and on the integral nature, in terms of perceptual structure [JS92], of the pan & zoom task. In particular, we were interested in comparing the following dimensions: bimanual vs. unimanual input; device-based vs. free-hand techniques; degrees of freedom (DOF) and associated kinesthetic and haptic feedback; and types of movements: linear gestures vs. circular, clutch-free gestures.

We made no *a priori* assumptions about relevant metaphors or technologies and considered freehand as well as device-based techniques. An extensive design and testing phase allowed

us to limit the number of candidates for the subsequent formal evaluation. For instance, the apparently intuitive solution that consists in using two hands or two fingers to zoom with pinch and stretch gestures was considered but quickly discarded: while these gestures work well on touch-sensitive surfaces such as tabletops, they are much less natural when performed in mid-air. Most importantly, they proved quite inaccurate, and tiring. We eventually identified a set of twelve candidate techniques. Their design was informed by related empirical studies reported in the literature and refined through prototyping and pilot testing. These techniques were organized according to three key dimensions forming a design space: unimanual vs. bimanual input, linear vs. circular gestures, and level of guidance through passive haptic feedback. In addition to performance (task time and accuracy), we took into account other usability issues, such as fatigue and ease of use.

We formulated hypotheses about these dimensions, and conducted a controlled experiment evaluating the twelve unique techniques generated from this design space. The task was a variation of Guiard *et al.*'s multiscale pointing task [GB04], adapted to take overshoots into account. Participants had to navigate through an abstract information space made of two groups of concentric circles separated from one another by a distance of up to 12 million pixels. They started at a high zoom level in one of the groups, zoomed out until the neighboring target group appeared, then panned and zoomed into that group until they had reached a position in the allowed range, i.e., at the correct zoom level and with the target correctly centered.

Our results identified several successful mid-air input techniques that can be used to navigate efficiently in very large datasets on wall-sized displays. In addition to identifying groups of alternative techniques based on performance, each with specific characteristics, the experiment also suggested clear results with respect to the factors that constituted the design space. For instance, despite their inherent and almost universal appeal, gestures performed in free space proved to be generally less efficient and more prone to fatigue than device-based input techniques. Adding guidance to input gestures increased, rather than decreased, accuracy. In accordance with the research literature, bimanual input techniques performed very well. Unimanual techniques performed honorably, and may still be considered in contexts of use where, for example, tools must be held in one hand to perform a domain/task specific action. A more surprising result was the generally higher efficiency of linear gestures when compared to circular, clutch-free gestures.

Addressing the Problem of Bezels on Tiled Displays

One of the major advantages of tiled monitor setups with respect to projection-based setups is the significantly higher pixel density. Using tiled monitors to build wall-sized displays actually has other advantages, including simpler setup and easier calibration. However, the resulting display walls suffer from the visual discontinuity caused by the bezels that frame each monitor. For some tasks, depending on the nature of the data being visualized, bezels can sometimes help users organize display space [Gru01, RCB⁺05, SHL03]. The grid formed by bezels can also help structure visual search [BBB10]. However, bezels are a problem when displaying large images such as maps or other visualizations that span multiple monitors. They create a

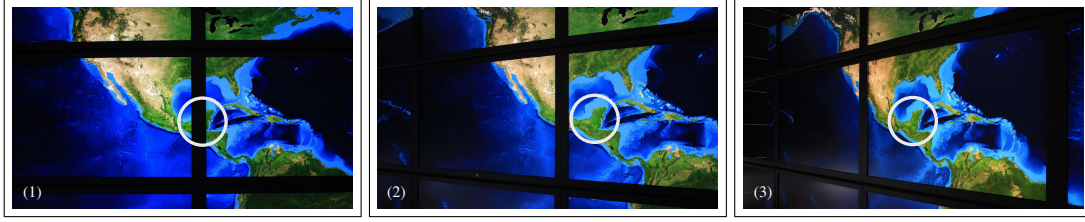


Figure 3.9 : GridScape: the grid formed by monitor bezels on wall displays is often compared to a french window. We designed two interaction techniques that transform that grid into an actual french window. On the map, (1) the Yucatán peninsula (white circle) is partially hidden by bezels. (2) With GridScape, users can reveal that part of the map simply by slanting their body or moving slightly to the right. (3) Moving further right, the entire eastern coast of Mexico can be shown without any bezel occlusion.

visual discontinuity, that can basically be treated in one of two ways [ETO⁺10]. The problem can be ignored entirely, displaying the picture as if monitors were juxtaposed seamlessly. This solution, called the *offset approach*, is simple and straightforward; it has been employed by many early platforms. However, since the panels do have seams, this method necessarily entails distortion of the rendered image, that will be proportional to the bezels' thickness. The other solution consists in taking the bezels into account [MH04a, RCB⁺05]. This solution, called the *overlay approach*, gives the overall impression that the bezels are a grid overlaid on top of a single image that spans the entire wall.

Neither of these approaches is ideal; they represent a trade-off. But to our knowledge these are the only two solutions in widespread use. One obvious way to address the problem would be to find a way to perfectly juxtapose the LCD panels. But bezels are unlikely to completely disappear soon. The ultra-thin bezels advertised in bleeding-edge products (as of early 2012) such as Samsung's 460UTN or LG's 47WV30 are still 6.9mm wide, which amounts to about 50 pixels at 100ppi, and come at the expense of resolution (1366x768 for a 47" diagonal), which does not qualify as ultra-high-resolution. Projection-based systems are inherently bezel-free, but are not a viable option, as they have a low pixel density, are difficult to align, and suffer more from problems such as color drift than LCD-based tiled displays. Some researchers have tried to mitigate the negative effect of bezels by removing the plastic casing of each panel [BN07, SADK⁺09]. However, those bezels cannot really be eliminated, but only halved due to technological constraints.

The overlay approach has often been presented using *french windows* as an analogy. We recently decided to investigate interaction techniques that draw upon this analogy, and make the display actually behave as if the visualization were observed through a french window. In [APPC12], we proposed two novel interaction techniques that let users reveal content hidden behind bezels. The first technique, called *ePan*, enables users to translate the entire image, displayed in overlay mode. Translation is controlled explicitly by interacting with a handheld

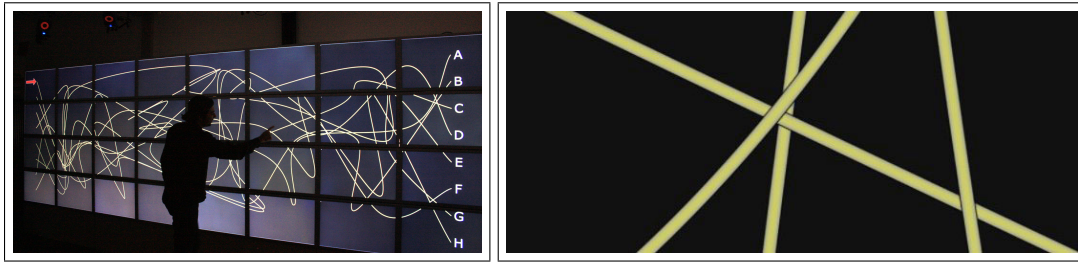


Figure 3.10 : (left) example path tracing trial. (right) details of the paths' rendering: using a stroke (thin black contour) and shadows helps disambiguate intersections.

device such as a smartphone. The second technique, called *GridScape*, adopts a more implicit approach by pushing the metaphor further: the visualization is offset in depth by putting some (virtual) distance between the LCD panels and the graphical projection plane. Then, using head tracking to simulate motion parallax [GGSF59], the technique enables users to see occluded parts of the image simply by moving in front of the display or changing their body posture, as illustrated in Figure 3.9.

While the techniques are conceptually easy to understand, implementing them on ultra-high-resolution wall displays requires distributing very large mipmaps on the multiple GPUs and computers that drive the wall, to eventually form a single coherent image. This image has to be rendered at a high frame-rate, typically 60fps, as otherwise the visual metaphor dies. The techniques were developed in C++ using Equalizer [EMP09b], a framework for OpenGL rendering on clusters of computers. In our implementation, the virtual canvas is made of a large rectangular mesh on which textures are applied. The techniques are implemented by translating that mesh orthogonally to (*GridScape*), or within (*ePan*), the wall display's plane. In *GridScape*, the camera's frustum adapts to the user's 3D head position and orientation. This information is obtained from WILD's motion tracking system.

To evaluate these techniques, we carefully operationalized visual path tracing tasks and conducted a controlled experiment that compared the performance and subjective preferences of users who had to follow paths of varying complexity across display tiles, as illustrated in Figure 3.10. Two techniques, *ePan* and a simple switch between offset and overlay modes, required explicit input from the user to reveal hidden content. They were compared to *GridScape* that relied, for that purpose, on users' physical navigation in front of the display. Results showed that for path tracing tasks of moderate complexity, both the technique that relies on explicit user actions (*ePan*) and the one that relies on more implicit information from users' physical navigation (*GridScape*) do improve performance compared to the basic overlay/offset techniques in use today. For more complex tasks, *ePan* has a clear advantage over all other techniques.

One limitation of all techniques studied is that they worked for a single user only. As mentioned earlier, wall displays are natural platforms for cooperative work, and we felt it was important that the techniques can function in scenarios involving multiple users. A user should be

able to activate a technique to look behind bezels near her without interfering with other users' activities, i.e., without changing the visualization in the regions the other users are focusing on.

A straightforward solution when manually switching between overlay and offset mode consists in toggling between overlay and offset modes only for the tiles located within the field of view of the user who is invoking the technique. Changes are applied locally, based on the user's position and orientation relative to the wall, and do not affect the other users' focus area – except for a user who would be standing back, looking at an overview of the entire collaborative workspace, but then such a user would arguably not be strongly impacted by those small-scale visual changes. This idea of per-user, locally-bounded visual transformations can also be adapted to the other two techniques. One solution for ePan is to apply a locally-bounded translation to the rectangular subregion corresponding to the user's visual focus area: the user can choose to trigger ePan in cooperative mode, which only translates that subregion (again computed from her position and head orientation) in the virtual canvas' plane, instead of translating the whole representation as the original ePan technique does. Using a continuous version of Stretchable Rubber Sheets [SSTR93], the immediate surroundings of this subregion get distorted (compressed or stretched depending on direction) to achieve a smooth transition with the other regions that form this user's context and the other users' focus area, that the transformation has left unaffected. Each user is free to invoke ePan independently, for her own purposes. A similar approach can be employed for GridScape, offsetting (in depth) only the subregion that is in the user's focus area. This case requires a more elaborate mapping, using fisheye-lens-like 1D distortion (Section 3.1.2) to smoothly integrate the locally transformed view into the surrounding context. Once offset in depth, the subregion's outer boundaries remain fixed, with only the offset plane and surrounding transition areas adapting to the user's head and body movements to simulate motion parallax.

3.2 Engineering of Multi-scale Interactive Systems

In their survey of user interface programming from 1992, Myers and Rosson reported that at this time an average of 48% of the code was devoted to programming the user interface part of applications [MR92]. As the level of sophistication of interfaces has grown significantly over the last two decades, this proportion is unlikely to have decreased, with the advent of post-WIMP interfaces [BL00, Bea00, HDD⁺04, JGH⁺08], multi-modal interaction [NC95, SJN08], the ever-growing capabilities of mobile devices and the novel types of interactive surfaces that can be combined together to create multi-surface applications [GKE⁺11, KB09].

On the desktop, a multitude of toolkits have been developed over the years, exploring different ways to address the complexity of user interface programming while enabling software developers to create increasingly elaborate graphical user interfaces: state machines [AB08, BB06], constrained-based programming [MMM⁺97], graphical compilation [TC11], instrumental interaction [Bea00, KB09], scene graphs [BGM04], possibly coupled with a dataflow model [HDD⁺04, DF04]. Some toolkits focus on specific types of visual interface components, such as information visualization toolkits [Fek04, HCL05] and zoomable user interface

toolkits [BGM04, **Pie05a**, PLVB00], while others try to be more generic [AB08, BB06, KB09, Lec03, MMM⁺97, TC11].

3.2.1 Easing the Development of Multi-scale User Interfaces

Before entering the PhD program at INRIA Rhône-Alpes and Xerox Research Centre Europe (XRCE) in 1999, I had been an intern at XRCE for 6 months, working on the *Visual Abstract Machine* (VAM), a user interface toolkit initiated by Jean-Yves Vion-Dury that enabled developers to create 2D zoomable user interfaces in Java and to program interaction through grammars that were specifying the allowed sequences of events and what sequences of user actions triggered what function calls in the application. When I started working on the development environment for the VXT visual programming language (Section 1.1), I decided to reuse and enhance the zoomable user interface part of this toolkit to develop the zoomable views that would enable XML developers to navigate in tree structures and transformation rules represented as uni-dimensional treemaps [**PVDQ01**, **VDP01**]. Jazz [BMG00] had not yet been published, and I was not aware that a full-fledged toolkit for developing ZUIs in Java was about to become available. I discarded the event grammar part of the VAM toolkit, which I found very intriguing but felt would require grammars of unmanageable complexity to capture all interaction. At that point, this stripped-down version of the VAM had no name, I had no plan for its development beyond making zoomable interface components for VXT, and it was not meant to become an actual UI toolkit. As I later decided to re-use and extend the capabilities of this component to develop the zoomable graph views of IsaViz during my internship and post-doc at MIT (Section 2.1), the toolkit eventually grew into a full-fledged zoomable user interface toolkit called ZVTM [**Pie05a**]. The toolkit has never been a research project *per se* but ever since it was made an open source project hosted on SourceForge in 2003, it has been used in many of my projects as well as by other team members and by third parties, either as an environment for prototyping novel interaction and visualization techniques and evaluating them [**ACP12**, **BPLL11**, **RJH11**, **ACP10**, **PBA10**, **CLP09**, **MCH⁺09**, **PA08**, **PAB07**, **Pie06**] or for the development of applications⁷. To date, the most ambitious project that makes use of ZVTM is the joint effort between ALMA and INRIA to develop advanced multi-scale visualization components for the ALMA radio-telescope, presented in more detail later in this section.

On one side, low-level graphics APIs such as OpenGL or Java2D are powerful but difficult to use, requiring the programmer to deal with low-level graphical operations and implementation problems. Writing components with these low-level APIs requires a lot of development and maintenance effort. On the other side, toolkits such as Swing are very powerful, generic and portable, but are limited to the conventional interface widgets of the WIMP model (*Windows Icons Menus and Pointers*). They cannot be used to develop advanced visualization and other post-WIMP interfaces.

⁷See <http://zvtm.sf.net> for representative examples.



Figure 3.11 : Multi-scale Navigation in Digital Libraries: ZUITS contains 578 research papers, typically 4-to-10 page long, all available as PDF documents. Users can navigate and read all documents using a zoomable user interface. Navigation can be organized by year of publication, by keywords represented as a tag cloud, or by authors.

The purpose of ZVTM is to give developers a solution situated at an intermediate level of abstraction. It aims at making the development of interface components involving complex structured graphics easier by hiding most of the complexity: performance, memory management, UI events, geometrical transformations, concurrent access. ZVTM focuses on high-quality visual rendering while maintaining good performance, and on the user experience by promoting foundational concepts such as *perceptual continuity* in graphical interfaces [RCM93] through a simple-to-use animation module. The toolkit is based on the metaphor of infinite universes called *virtual spaces* that can be observed through movable and zoomable *cameras*. Virtual spaces contain graphical objects called *glyphs*: geometrical shapes, bitmap images, text. All glyphs rely on the same polymorphic object model. A glyph belongs to a specific virtual space, but can be observed through different cameras simultaneously as each virtual space can contain multiple cameras. Cameras are associated with viewports called *views* which correspond to rectangular components (windows, or embedded canvases) in the user interface. Various interaction techniques are implemented on top of this basic set of core concepts, and can easily be combined: smooth pan & zoom navigation [CKB08], superimposed translucent layers [Lie94], magnification lenses from the Sigma Lens framework [PA08, PBA10, ACP10], bird's eye view [CKB08], rate-based scrolling, speed-dependent automatic zooming [IH00], topology-aware network navigation techniques [MCH⁺09], pointing facilitation techniques [CLP09].

The toolkit supports both geometric and semantic zooming [PF93], enabling the development of user interfaces to navigate in very large information spaces. For instance, it was used

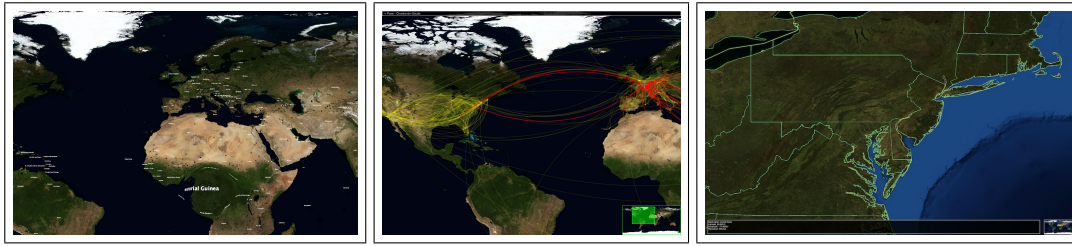


Figure 3.12 : Navigating in a multi-scale version of NASA’s Blue Marble Next Generation world map [SVS⁺05]: 86 400 x 43 200 pixels decomposed into 2 730 tiles arranged as a pyramid. The map is enriched with geographical data taken from the Geonames database, the worldwide air traffic network, and boundaries for countries and administrative regions publicly available as ESRI shapefiles.

to write the ZUIST application⁸, demonstrated in Newport, RI, for the 20th anniversary of the ACM UIST conference in 2007. The application, illustrated in Figure 3.11, lets users navigate in all papers published during the conference’s first 20 years using a zoomable user interface. It was developed quickly thanks to ZVTM and its multi-scale/semantic zooming engine: the source code consists of only 2400 lines of Java code, plus 1200 lines of Python code that were written first, to parse the ACM Digital Library’s metadata and produce the multi-scale scene structure that organizes all 578 PDF documents. We later wrote a similar application for Université Paris-Sud’s computer science laboratory, that lets users navigate in all documents published from 2005 to 2008, ranging from book chapters, journal and conference papers to PhD dissertations and technical reports, for a total of 1,500 documents. A PDF version of the document was available for 753 of them, amounting to about 15,000 pages actually readable directly within the application. The figures in terms of source code size are similar to those of ZUIST. Figure 3.12 illustrates a very different application: a prototype geographical information system that can query Web services to overlay geolocated information items, including complex air traffic networks with support for topology-aware navigation techniques [MCH⁺09], on top of NASA’s Blue Marble Next Generation map and ESRI shapefiles. That particular application’s source consists of 3000 lines of Java code. All of these applications ran (and still run) smoothly on mid-range machines (laptops or workstations), and did not require advanced hardware resources.

The toolkit has also been used to write interface components for collaborative research projects, including ANR Blanc project Holyrisk⁹, in which our contribution is to develop an application that lets domain experts navigate and read large collections of annotated docu-

⁸<http://www.acm.org/uist/archive/uist2.0/ZUIST.html>

⁹Collaborative project involving Met@risk (agricultural research, INRA), University of California in San Diego’s sociology department, In-Situ (situated interaction, INRIA, CNRS & Université Paris-Sud), EFSA (European Food Safety Agency) and JIFSAN (food safety, US Food & Drug Administration, University of Maryland). Principal Investigator for partner INRIA: Emmanuel Pietriga. More information at http://www.paris.inra.fr/metarisk/research_unit/research_projects/holyrisk

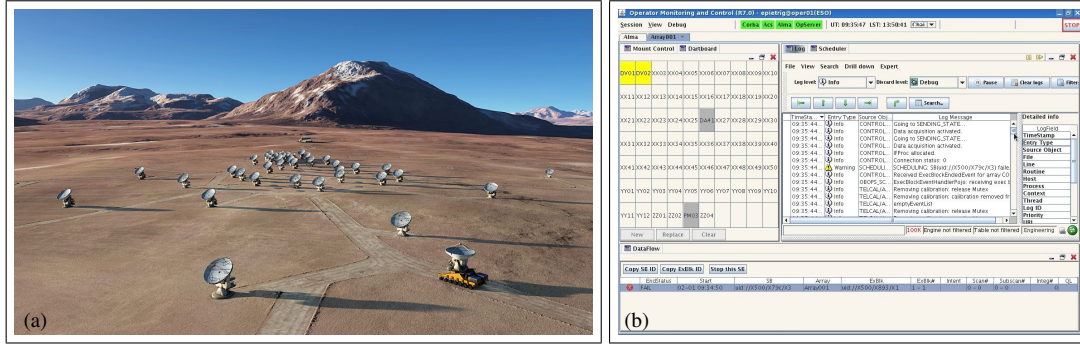


Figure 3.13 : (a) Artist rendering of the Atacama Large Millimeter/submillimeter Array, in an extended configuration © ALMA (ESO/NAOJ/NRAO)/L. Calçada (ESO). (b) Example WIMP components of the original operations monitoring and control software.

ments related to food risk assessment and structured into a database. But as mentioned earlier, the most ambitious project involving ZVTM to date is the joint effort between ESO¹⁰, NRAO¹¹ and INRIA to develop advanced visualization components for the operations monitoring and control software used to run the ALMA radio-telescope¹² currently under construction in the Chilean Andes (Figure 3.13-a). This ongoing project [SPSG10, SPSS11, PCS+12], that started in late 2009, is about the design and implementation of novel multi-scale user interface components targeted at both operators and astronomers that will be in charge of running observations from the instrument's control room.

This project started after experience operating the telescope in the early commissioning phase had shown that the graphical operator interface implemented with the WIMP toolkit Java Swing would not scale well from the 8 antennas that were on-site at the time to the 66 under construction. Conventional WIMP widgets and the WIMP paradigm itself were not powerful enough (Figure 3.13-b), and more advanced visualization components, defined through a user-centered design approach, had to be implemented. Figure 3.14 shows some examples of UI components already implemented, and for some of them currently deployed and tested in the control room. The design of those components is informed by a series of participatory design workshops held on site at the Operations Support Facility regularly since late 2009. The goal of these new components and capabilities, that include a general synchronization mechanism to easily coordinate multiple views [NS00], is to enable operators and astronomers to identify trouble spots and react to failures quickly in an environment where situation awareness [EJ12] is a critical element to the safe and efficient functioning of the observatory.

¹⁰European Southern Observatory, <http://www.eso.org>

¹¹National Radio Astronomy Observatory, <http://www.nrao.edu>

¹²Atacama Large Millimeter/submillimeter Array, <http://almaobservatory.org>

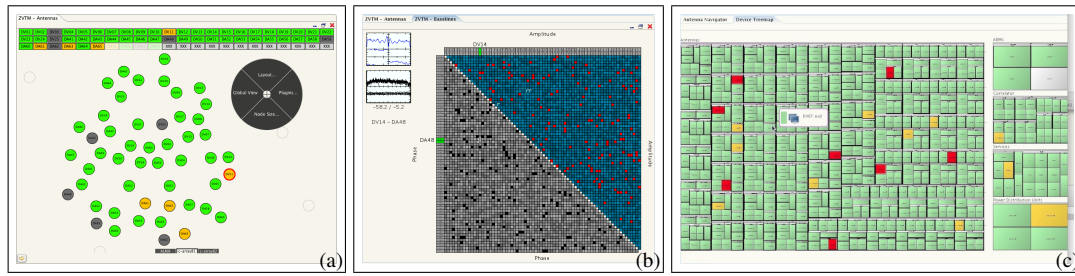


Figure 3.14 : (a) Map of all 66 antennas in an interface that includes, among other features, geographical coordinate distortion to optimize the representation depending on the physical configuration of antennas on site, semantic zooming [PF93] and control menus [PLVB00]. The amount of information about individual antennas ranges from a simple solid fill color giving a high-level overview of system status (as shown here) to a detailed block diagram at highest zoom level. (b) Adjacency matrix [HFM07] giving information about phase and amplitude for the baselines that link antennas pairwise to form an almost fully connected graph. (c) Zoomable treemap [BJH99, BL07] representation of all devices' status, including antennas, the correlator, and supporting hardware.

3.2.2 A User Interface Toolkit for Cluster-Driven Wall-sized Displays

While ZVTM was originally aimed at the development of desktop post-WIMP interfaces, we started working in 2009, together with Romain Primet, research engineer at INRIA, on an extension to the toolkit that would enable applications implemented on top of ZVTM to run on ultra-high-resolution wall-sized displays such as WILD (Section 3.1.4) with very few modifications.

This on-going effort is motivated by the fact that research on cluster-driven wall displays has mostly focused on techniques for parallel rendering of complex 3D models. There has been comparatively little research effort dedicated to other types of graphics and to the software engineering issues that arise when prototyping novel interaction techniques or developing full-featured applications for such displays. Generally-speaking, visualization platforms such as WILD pose new research challenges. From a *computer graphics perspective*: how to render complex graphics at high frame rates, taking advantage of the cluster's computing and rendering power. From a *human-computer interaction perspective*: how to design effective visualizations that take advantage of the specific characteristics of large, ultra-high-resolution surfaces; how to design interaction techniques that are well-adapted to this particular context of use, and how to handle the multiple and heterogeneous input devices and modalities typically used in this context. Finally, from a *software engineering perspective*: how to enable the rapid prototyping, development, testing and debugging of interactive applications running on clusters of computers, providing the right abstractions.

Work on the zvtm-cluster extension to ZVTM, and on the jBricks toolkit that now includes it, focuses on providing an answer to the latter question, that we consider essential to foster more research and development from the HCI perspective.

While there has been much research effort put into the development of solutions to enable the efficient display of 3D graphics on tiled displays driven by clusters of computers over the last ten years [HHN⁺02, EMP09a, NSS⁺06], not all wall display applications use 3D graphics. With the introduction of ultra-high resolution, high-quality 2D graphics open wall-sized displays to new applications, e.g., in astronomy (Figure 3.15-b), geospatial intelligence and visual analytics at large, as discussed earlier. These applications essentially combine very large bitmap images, high-quality text and 2D vector graphics, e.g., satellite imagery augmented with data layers, or information visualization techniques for the display of large datasets, e.g., for the visual exploration of large networks (Figure 3.15-a). However, existing solutions for distributed rendering on clusters, which are essentially low-level 3D graphics APIs based upon OpenGL, are not suited to high-quality 2D graphics. They work well for the high-performance visualization of textured 3D scenes, but are ill-suited to programming rich 2D graphics interfaces, lacking appropriate support for the management and efficient rendering of text, line styles, arbitrary 2D shapes and WIMP widgets. This was already observed for desktop application programming [BM98], and remains true for cluster-driven wall-displays.

To address this need, we developed jBricks, a Java toolkit for the development of post-WIMP applications executed on cluster-driven wall displays, that integrates zvtm-cluster and a versatile input management module into a coherent framework hiding low-level details from the developer [PHNP11] (*Rapid Development of User Interfaces on Cluster-Driven Wall Displays with jBricks*, available in Appendix A). The goal of this framework is to ease the development, testing and debugging of interactive visualization applications. It also offers an environment for the rapid prototyping of novel interaction techniques and their evaluation through controlled experiments (Figure 3.15-c), such as the one we conducted about mid-air pan-and-zoom techniques for wall-sized displays, summarized in Section 3.1.4 and detailed in [NWP⁺11].



Figure 3.15 : jBricks applications running on the WILD platform. (a) Zoomed-in visualization of the North-American part of the world-wide air traffic network overlaid on NASA's Blue Marble Next Generation images augmented with country borders ESRI shapefiles. (b) Panning and zooming in Spitzer's Infrared Milky Way (4.7 gigapixels). (c) Controlled laboratory experiment for the evaluation of mid-air multi-scale navigation techniques (Section 3.1.4).

Rich interactive 2D desktop applications, usually termed post-WIMP applications, are typically developed with structured graphics toolkits [AB08, BGM04, HDD⁺04, **Pie05a**] that provide useful abstractions on top of low-level APIs. They enable rapid prototyping and development of advanced interactive visualizations. By developing jBricks, our goal was to offer a structured graphics toolkit capable of running transparently on cluster-driven wall displays and capable of handling a wide range of input devices and modalities. From a graphics perspective, this requires hiding the complexity entailed by having to distribute rendering on multiple computers. While our focus was on expressiveness and ease-of-use, we also paid attention to scalability issues, adapting ideas originally developed for efficient distributed 3D rendering to our context, such as the use of a multicast protocol to transmit updates to cluster nodes, and a culling algorithm adapted to zoomable user interfaces. From an input management perspective, this required going beyond the basic redirection mechanisms found in existing distributed rendering frameworks that only support conventional input devices, i.e., mouse and keyboard operated from the master computer. Support for other devices had mostly been achieved *via* ad hoc solutions (drivers or libraries) that were strongly integrated and statically linked within applications. This approach was not generic and flexible enough when exploring and prototyping novel interaction techniques [DF04]. We chose an alternative approach, that consisted in providing high-level abstractions of input modalities that enable association and runtime substitution of devices, and that had proven successful in other domains, including physical ubiquitous computing, virtual reality and in the more general context of post-WIMP applications.

jBricks, which was written together with Stéphane Huot and Romain Primet, has now reached a level of maturity that has enabled members of the WILD project to use it for various purposes, ranging from the development of prototype applications such as the one developed in collaboration with astrophysicists from the Institut d'Astrophysique Spatiale for the visualization of large FITS images, to multiple controlled experiments, and demonstrations of interactive navigation in very large information spaces such as Spitzer's image of the inner part of our galaxy (Figure 3.15-b) or the 26 gigapixel panorama of Paris. Other research laboratories equipped with wall-sized displays are also considering using the toolkit, which we have made available as an open-source projet.

Perspectives

The two main themes around which I have organized my research over the last ten years are closely related and cross-fertilize one another. Multi-scale user interfaces are essential to the management and understanding of massive datasets. Interlinked datasets enriched with even a small amount of semantics have the potential to help create better user interfaces, by providing users with more relevant query results and giving them efficient means to navigate and relate those results. However, there are still many research questions and problems to address in both themes, and at their intersection to further foster cross-fertilization. My research agenda for the coming years, presented in the following sections, is positioned in this context, with a focus on critical systems that require a high level of situation awareness and efficient access to the data. Those are essential elements to support the decision making process and implementation of actions in a timely manner [EJ12], taking into account the often cooperative nature of the work and the resulting need for task coordination [BHR⁺92, BN99, JW01, Mac99]. Examples of such work contexts include air traffic control, crisis and emergency management centers, control rooms of power plants and large scientific instruments such as telescopes. The new research and innovation center that INRIA is creating in Chile will give me the opportunity to conduct research in this direction, as detailed below.

4.1 Interacting with Massive Webs of Data

Most efforts in terms of user interface design and development for Semantic Web data, including my own, have so far focused on tools for software developers and for experts that create ontologies and populate them. More end-user-oriented tools are starting to appear, such as the so-called *linked data browsers*, some of which make use of presentation knowledge expressed using the Fresnel vocabulary (Section 2.2). However, those browsers are in most cases very crude ways of navigating Webs of Data, based on point and click WIMP interfaces, or hyperlink interfaces that present data to users in a very basic page-centric Web-of-documents manner.

It is not clear yet how end-users will interact with the Web of Data [Hea08], but to be successful, interaction paradigms that let users navigate that new Web will have to be tailored to this radically different way of browsing information. Now that building blocks are in place, from knowledge representation and query languages all the way up to the expression of data presentation knowledge [PBKL06] and to mechanisms like look-up services [TDO07] and spreading activation [DLK⁺10], we can start investigating the user interface part of the *Semantic Web layer cake*, an area of research and development that has not received much attention so far [Hen12].

It is unlikely that one type of browser, or even one representation and navigation paradigm, can effectively support all user tasks and suit all types of data, especially when dealing with large collections of interlinked datasets. The question at this point is actually not to define a generic paradigm, but rather to integrate navigation techniques and basic representation paradigms in a coherent framework, understanding their strengths and weaknesses through more ambitious user studies than the few experiments that have been run so far. Core paradigms

and techniques exist, such as faceted browsing [HvOH06, ODD06], graphical representations [BPLL11, Pie06, TXZ⁺05, WP06], rich interface clients [QHK03, sSO⁺05], but they all serve different purposes, and none of these is sufficient on its own. A radical shift might be needed, that departs from the path that consists in adapting and applying techniques developed for other purposes to the Web of data, possibly adopting a *thing-centric* [Hea08] or *resource-centric* way of presenting and navigating in the data, but that would go far beyond what tools like the Tabulator [BLCC⁺06] currently supports.

Such a solution could for instance couple *multi-scale* and *thing-centric* navigation. At any given time, one thing (i.e., one resource) would be the focus of attention. Users could navigate from resource to resource, fetching relevant items dynamically, depending on context [DLK⁺10], leaving a trail of past explorations on the infinite canvas of a zoomable user interface component, with elaborate mechanisms to backtrack, and to relate, compare and contrast the current resource with previously explored ones. Taking advantage of the advanced graphics capabilities of ZUIs [Pie05a], users could ask for data items, and collections thereof, to be presented in a particular manner, e.g., be plotted on an interactive map, represented as a network of connected things [BPLL11], possibly with style sheets applied to them [Pie06], or summarized in statistical charts. Zooming in and out from the focused resource, and from the trail of items explored so far, would change the level of detail conveyed to users thanks to semantic zooming [PF93].

Semantic Web technologies enable users to effortlessly mashup *data*, somewhat like some Web services enable users to create mashups on the Web of documents [AGM08, ZR08]. A platform implementing the navigation paradigm envisioned here would likely provide a powerful user interface layer for exploring those data mashups. Coupled with fast input techniques based on keystrokes for issuing commands, similar to what add-ons like Quicksilver for Mac OS X [QSA12] enable, such a solution would also be potentially interesting as an interface paradigm for efficiently troubleshooting critical systems.

4.2 Human-Computer Interaction and Situation Awareness

Data enriched with (even a small dose of) semantics should play a major role in future critical systems that support tasks related to crisis and emergency management activities, as well as systems that deal with vast amounts of data, such as the large scientific instruments mentioned earlier. (Semi)structuring the data, adding semantics to them, will facilitate access and understanding from a user's perspective: query results will be more relevant, datasets will be easier to relate and merge thanks to interlinking, thus supporting more efficient troubleshooting and decision making. In other words, achieving time-critical tasks should become more efficient as the quality and richness of the data, in terms of structure and machine-processable semantics, grow. Advances in this area are crucial for enhancing information collection, sharing and dissemination, as emphasized in the Pacific Northwest National Laboratory's research agenda for *Precision Information Environments* (PIE) [Pac11]. However, we are still far from achieving the environments envisioned in the video produced last year by that same laboratory [Pac10].

There are clear gaps, and research opportunities in many fields beyond data management, one of them being human-computer interaction.

Central to these contexts of work is the notion of situation awareness, i.e., how workers perceive elements of the environment with respect to time and space, how they comprehend their meaning, and the mental models that help them predict future states of those elements. Situation awareness must be supported by efficient computing systems and their associated user interfaces, whose design should be primarily driven by user-centered design methods. User interface technologies always have and will continue to play a central role in the performance and efficiency of critical systems. Novel user interface paradigms and technologies such as gesture-based interaction, communication systems for computer-supported cooperative work, mixed reality systems, as well as multi-surface environments including tablets, portable digital assistants and ultra-high-resolution wall-sized displays are all likely to have a strong impact in this domain.

However, several areas of research are still in their infancy. For instance, while there is already a significant body of work on the visualization of massive datasets on wall-sized displays, we have barely scratched the surface of that topic, and for now, those platforms remain mostly confined to research laboratories. While we are making progress on all fronts, more research and development is needed in computer graphics and software engineering, but most importantly from the *interaction design* perspective. Questions about basic interactions such as remote pointing at targets have been thoroughly researched (see [NPBL11] for an overview), techniques for more complex tasks such as multi-scale navigation have been designed and evaluated, concepts such as physical navigation have been developed in relation to visual acuity, and some researchers have started to look into the problem of collaborative interaction in these environments. But we are still far from *walk-up-and-use* systems, or even from actually achieving fluid interaction with wall-sized displays.

Questions to address include, for instance, collaborative work in multi-scale information spaces. The techniques studied in [NWP⁺11] enable one or more persons to pan & zoom in mid-air in front of a wall display. But they do not address the problem of how to partition that display when multiple users want to explore different regions of the information space (e.g., a world map). Simply splitting the display in discontinuous juxtaposed viewports would be trivial. But this would not provide much support for cooperative work. Multi-scale user interface schemes initially developed for the desktop such as focus+context and overview+detail techniques could help make users aware of, and relate to, each others' position. Beyond a mere transposition of desktop techniques to wall displays, solutions could take advantage of the significantly higher pixel density of the latter to rethink how these interface schemes can be integrated: overview+detail and focus+context could for instance be more deeply, possibly recursively, nested, with elaborate transitions between different levels of detail [PBA10] helping users get a clear picture of *who is working on what*.

Another example of question to address is that of seamless manipulation of content items on multiple heterogeneous surfaces. Gesture-based interfaces are natural candidates for this and seem to actually be interesting solutions, at least in some contexts, but even if technologies such as Oblong's g-speak platform have significantly advanced the state-of-the-art in that domain [ZBL⁺10], there is still a lot of room for improvements. Yet another question is that of taking advantage of the specific characteristics of ultra-high-resolution displays to create *perceptually multi-scale* visualizations, coupling physical navigation with visual aggregation, a perceptual phenomenon that naturally occurs when the conditions in terms of pixel density and user position with respect to the display entail that the fine details of the representation are beyond the user's visual acuity. A few studies have started to look at the potential of patterns that arise as a result of visual aggregation [EALN11, YHN07], but much more research needs to be done in that area as well.

Finally, very few real-world applications have been developed so far for these environments. Most software running on these platforms are little more than programs for running controlled experiments or demonstrations which, even if quite elaborate, offer only a few features and are far from full-fledged applications, that would require exposing many more commands through the user interface. Even ports of existing applications such as Google Earth based on Chromium [HHN⁺02] or Liquid Galaxy [Goo11] cannot really be considered real-world applications, as they do not expose the wealth of features of the original desktop version in a way that would make them actually usable on wall displays. Our experience with the WILD project, working with scientists from other disciplines on prototype domain-specific visualization applications [BCE⁺12], has shown that designing and implementing full-featured applications that domain experts can use to conduct actual work in the real-world is far from trivial and requires significant interaction design and software development effort.

Joining the *Communication and Information Research and Innovation Center*¹ in Santiago de Chile in July 2012 will give me the opportunity to start out a new team, *Massive Data*, with expertise in human-computer interaction, classification and Semantic Web technologies. The team's research and development activities will focus on the questions introduced above. They will include:

- the continuation of the work started in 2010 on user-centered design and implementation of operations monitoring and control user interfaces for the ALMA radio-observatory's control room (Section 3.2.1);
- the design and implementation of tools to support research in astronomy beyond the work done in ALMA's control room, such as interfaces based on classification and clustering techniques for, e.g., browsing large collections of galaxies stored in massive surveys like SDSS² and the planned LSST³, whose data will be made public;

¹CIRIC is a joint research and innovation center created in March 2012 by INRIA and six Chilean universities.

²Sloan Digital Sky Survey, <http://www.sdss.org>

³Large Synoptic Survey Telescope, <http://www.lsst.org>

- the design and evaluation of visualization techniques for massive datasets on large ultra-high-resolution displays, and the implementation of domain-specific applications for, e.g., communication network monitoring and metrology and possibly other domains such as the mining industry;
- the design and implementation of user interfaces for browsing data repositories such as those made available by governments in the context of various *open data* initiatives.

Bibliography

*Note: references in **bold type** are articles that I authored or co-authored.*

- [AB08] Caroline Appert and Michel Beaudouin-Lafon. Swingstates: adding state machines to java and the swing toolkit. *Softw. Pract. Exper.*, 38:1149–1182, September 2008.
- [ABM05] Caroline Appert, Michel Beaudouin-Lafon, and Wendy E. Mackay. Context matters: Evaluating interaction techniques with the cis model. *British HCI - People and Computers XVIII—Design for Life*, pages 279–295, 2005.
- [ABS99] Serge Abiteboul, Peter Buneman, and Dan Suciu. *Data on the Web: From Relations to Semistructured Data and XML*. Morgan Kaufmann, 1999.
- [Ack89] Russell L Ackoff. From data to wisdom. *Journal Of Applied Systems Analysis*, 16(1):3–9, 1989.
- [ACP10] Caroline Appert, Olivier Chapuis, and Emmanuel Pietriga. High-precision Magnification Lenses. In *CHI '10: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 273–282, 2010.
- [ACP12] Caroline Appert, Olivier Chapuis, and Emmanuel Pietriga. Dwell-and-Spring: Undo for Direct Manipulation. In *CHI '12: Proceedings of the 30th international conference on Human factors in computing systems*, pages 1957–1966. ACM, 2012.
- [AGM08] Serge Abiteboul, Ohad Greenshpan, and Tova Milo. Modeling the mashup space. In *Proceedings of the 10th ACM workshop on Web information and data management, WIDM '08*, pages 87–94. ACM, 2008.
- [AMM⁺08] Wolfgang Aigner, Silvia Miksch, Wolfgang Müller, Heidrun Schumann, and Christian Tominski. Visual Methods for Analyzing Time-Oriented Data. *IEEE Transactions on Visualization and Computer Graphics*, 14(1):47–60, 2008.
- [APPC12] Rodrigo Andrade de Almeida, Clément Pillias, Emmanuel Pietriga, and Pierre Cubaud. Looking behind bezels: French windows for wall displays. In *AVI '12: Proceedings of the working conference on Advanced visual interfaces*, pages 124–131. ACM, 2012.
- [BAP⁺05] Paolo Buono, Aleks Aris, Catherine Plaisant, Amir Khella, and Ben Shneiderman. Interactive Pattern Search in Time Series. In *Proceedings of Visualization and Data Analysis*, pages 175–186, 2005.

- [BB06] Renaud Blanch and Michel Beaudouin-Lafon. Programming rich interactions using the hierarchical state machine toolkit. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '06, pages 51–58. ACM, 2006.
- [BBB⁺95] Margaret M. Burnett, Marla J. Baker, Carisa Bohus, Paul Carlson, Sherry Yang, and Pieter van Zee. Scaling up visual programming languages. *Computer*, 28:45–54, March 1995.
- [BBB10] Xiaojun Bi, Seok-Hyung Bae, and Ravin Balakrishnan. Effects of interior bezels of tiled-monitor large displays on visual search, tunnel steering, and target selection. In *Proceedings of the 28th international conference on Human factors in computing systems*, pages 65–74. ACM, 2010.
- [BBLP11] David Beckett, Tim Berners-Lee, and Eric Prud'hommeaux. Turtle: Terse RDF Triple Language (W3C Working Draft). <http://www.w3.org/TR/2011/WD-turtle-20110809/>, August 2011.
- [BCE⁺12] Michel Beaudouin-Lafon, Olivier Chapuis, James Eagan, Tony Gjerlufsen, Stéphane Huot, Clemens Klokmoose, Wendy Mackay, Mathieu Nancel, Emmanuel Pietriga, Clément Pillias, Romain Primet, and Julie Wagner. Multi-surface Interaction in the WILD Room. *IEEE Computer*, 2012. To appear.
- [Bea00] Michel Beaudouin-Lafon. Instrumental interaction: an interaction model for designing post-wimp user interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '00, pages 446–453, New York, NY, USA, 2000. ACM.
- [Bec04] Dave Beckett. RDF/XML Syntax Specification (Revised), W3C Recommendation. <http://www.w3.org/TR/rdf-syntax-grammar/>, February 2004.
- [BGM04] Benjamin B. Bederson, Jesse Grosjean, and Jon Meyer. Toolkit design for interactive structured graphics. *IEEE Transactions on Software Engineering*, 30(8):535–546, 2004.
- [BHJ99] Mark Bruls, Kees Huizing, and JarkevanWijk. Squarified treemaps. In *In Proceedings of the Joint Eurographics and IEEE TCVG Symposium on Visualization*, pages 33–42, 1999.
- [BHR⁺92] Richard Bentley, John A. Hughes, Dave Randall, Tom Rodden, Pete Sawyer, Dan. Shapiro, and Ian Sommerville. Ethnographically-informed systems design for air traffic control. In *Proceedings of the 1992 ACM conference on Computer-supported cooperative work*, CSCW '92, pages 123–129. ACM, 1992.
- [BL00] Michel Beaudouin-Lafon and Henry Michael Lassen. The architecture and implementation of cpn2000, a post-wimp graphical application. In *Proceedings of the 13th annual ACM symposium on User interface software and technology*, pages 181–190. ACM, 2000.
- [BL05] Tim Berners-Lee. Primer: Getting into RDF & Semantic Web using N3. <http://www.w3.org/2000/10/swap/Primer.html>, 2005.
- [BL07] Renaud Blanch and Eric Lecolinet. Browsing zoomable treemaps: Structure-aware multi-scale navigation techniques. *IEEE Transactions on Visualization and Computer Graphics*, 13:1248–1253, November 2007.
- [BL09] Tim Berners-Lee. Ted talk: Tim berners-lee on the next web, 2009. http://www.ted.com/talks/tim_berniers_lee_on_the_next_web.html.
- [BLCC⁺06] Tim Berners-Lee, Yuhsin Chen, Lydia Chilton, Dan Connolly, Ruth Dhanaraj, James Hollenbach, Adam Lerer, and David Sheets. Tabulator: Exploring and Analyzing linked data on the Semantic Web. In *Proceedings of the 3rd Int. Semantic Web User Interaction Workshop*, Athens, USA, November 2006.

- [BLFM05] Tim Berners-Lee, Roy T. Fielding, and Larry Masinter. Uniform Resource Identifier (URI): Generic Syntax. IETF, <http://www.ietf.org/rfc/rfc3986.txt>, January 2005.
- [BLP05] Chris Bizer, Ryan Lee, and Emmanuel Pietriga. Fresnel - A Browser-Independent Presentation Vocabulary for RDF. In *Second International Workshop on Interaction Design and the Semantic Web*, October 2005.
- [BLP10] Benjamin Bach, Gennady Legostaev, and Emmanuel Pietriga. Visualizing Populated Ontologies with OntoTrix. In *Proceedings of the ISWC 2010 Posters and Demonstrations Track: Collected Abstracts*, pages 85–88, 2010.
- [BM86] William Buxton and Brad Myers. A study in two-handed input. *SIGCHI Bull.*, 17(4):321–326, 1986.
- [BM97] Ravin Balakrishnan and I. Scott MacKenzie. Performance differences in the fingers, wrist, and forearm in computer input control. In *CHI '97: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 303–310. ACM, 1997.
- [BM98] Ben Bederson and Jon Meyer. Implementing a zooming user interface: experience building pad++. *Softw. Pract. Exper.*, 28:1101–1135, August 1998.
- [BMG00] Benjamin B. Bederson, Jon Meyer, and Lance Good. Jazz: an extensible zoomable user interface graphics toolkit in java. In *Proceedings of the 13th annual ACM symposium on User interface software and technology*, UIST '00, pages 171–180. ACM, 2000.
- [BN99] Johan Berndtsson and Maria Normark. The coordinative functions of flight strips: air traffic control work revisited. In *Proceedings of the international ACM SIGGROUP conference on Supporting group work*, GROUP '99, pages 101–110. ACM, 1999.
- [BN07] Robert Ball and Chris North. Visual analytics: Realizing embodied interaction for visual analytics through large displays. *Comput. Graph.*, 31:380–400, June 2007.
- [BNB07] Robert Ball, Chris North, and Doug Bowman. Move to improve: promoting physical navigation to increase user performance with large displays. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 191–200. ACM, 2007.
- [BPLL11] Benjamin Bach, Emmanuel Pietriga, Ilaria Liccardi, and Gennady Legostaev. OntoTrix: a Hybrid Visualization for Populated Ontologies. In *WWW '11: Proceedings of the 20th international conference companion on World wide web*, pages 177–180. ACM, 2011.
- [BSP⁺93] Eric A. Bier, Maureen C. Stone, Ken Pier, William Buxton, and Tony D. DeRose. Tool-glass and magic lenses: the see-through interface. In *SIGGRAPH '93: Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, pages 73–80. ACM, 1993.
- [CCF95] M. Sheelagh T. Carpendale, David J. Cowperthwaite, and F. David Fracchia. 3-dimensional pliable surfaces: for the effective presentation of visual information. In *UIST '95: Proc. ACM Symp. on User Interface Software and Technology*, pages 217–226. ACM, 1995.
- [CCF97] M. Sheelagh T. Carpendale, David J. Cowperthwaite, and F. David Fracchia. Making distortions comprehensible. In *VL '97: Proc. of the 1997 IEEE Symp. on Visual Languages*, page 36. IEEE, 1997.

- [CD99] James Clark and Steve DeRose. XML Path Language (XPath) version 1.0. <http://www.w3.org/TR/xpath>, November 1999.
- [CFP06] Sarah Cohen-Boulakia, Christine Froidevaux, and Emmanuel Pietriga. Selecting Biological Data Sources and Tools with XPR, a Path Language for RDF. In *Pacific Symposium on Biocomputing (PSB)*, Maui, Hawaii, pages 116–127, January 2006.
- [CGGG⁺05] Ana Conesa, Stefan Götz, Juan Miguel Garcia-Gomez, Javier Terol, Manuel Talon, and Montserrat Robles. Blast2GO: a universal tool for annotation, visualization and analysis in functional genomics research. *Bioinformatics*, 21(18):3674–3676, 2005.
- [CKB08] Andy Cockburn, Amy Karlson, and Benjamin B. Bederson. A review of overview+detail, zooming, and focus+context interfaces. *ACM Computing Surveys*, 41(1):1–31, 2008.
- [CLP09] Olivier Chapuis, Jean-Baptiste Labrune, and Emmanuel Pietriga. DynaSpot: Speed-Dependent Area Cursor. In *CHI '09: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 1391–1400. ACM, 2009.
- [CM01] M. Sheelagh T. Carpendale and Catherine Montagnese. A framework for unifying presentation space. In *UIST '01: Proc. ACM Symposium on User Interface Software and Technology*, pages 61–70. ACM Press, 2001.
- [CMS99] Stuart K. Card, Jock D. Mackinlay, and Ben Shneiderman. *Readings in information visualization: using vision to think*. Morgan Kaufmann, 1999. 712 pages.
- [CVBC08] Géry Casiez, Daniel Vogel, Ravin Balakrishnan, and Andy Cockburn. The impact of control-display gain on user performance in pointing tasks. *Human-Computer Interaction*, 23(3):215–250, 2008.
- [DF04] Pierre Dragicevic and Jean-Daniel Fekete. Support for input adaptability in the icon toolkit. In *ICMI '04: Proceedings of the 6th international conference on Multimodal interfaces*, pages 212–219. ACM, 2004.
- [DLK⁺10] Alan Dix, Giorgos Lepouras, Akrivi Katifori, Costas Vassilakis, Tiziana Catarci, Antonella Poggi, Yannis Ioannidis, Miguel Mora, Ilias Daradimos, Nazihah Md.Akim, Shah Rukh Humayoun, and Fabio Terella. From the web of data to a world of action. *Web Semantics*, 8:394–408, November 2010.
- [DS05] Martin Dürst and Michel Suignard. Internationalized Resource Identifiers (IRIs). IETF, <http://www.ietf.org/rfc/rfc3987.txt>, January 2005.
- [EALN11] Alex Endert, Christopher Andrews, Yueh Hua Lee, and Chris North. Visual encodings that support physical navigation on large displays. In *Proceedings of Graphics Interface 2011*, GI '11, pages 103–110. Canadian Human-Computer Communications Society, 2011.
- [EGK⁺01] John Ellson, Emden R. Gansner, Eleftherios Koutsofios, Stephen C. North, and Gordon Woodhull. Graphviz - open source graph drawing tools. In *Graph Drawing*, pages 483–484, 2001.
- [EJ12] Mica R. Endsley and Debra G. Jones, editors. *Designing for Situation Awareness: an Approach to User-Centered Design*. CRC Press, Taylor & Francis, January 2012. 370 pages.
- [EMP09a] Stefan Eilemann, Maxim Makhinya, and Renato Pajarola. Equalizer: A Scalable Parallel Rendering Framework. *IEEE Transactions on Visualization and Computer Graphics*, 15(3):436–452, 2009.

- [EMP09b] Stefan Eilemann, Maxim Makhinya, and Renato Pajarola. Equalizer: A scalable parallel rendering framework. *IEEE Transactions on Visualization and Computer Graphics*, 15(3):436–452, 2009.
- [ETO⁺10] Achim Ebert, Sebastian Thelen, Peter-Scott Olech, Joerg Meyer, and Hans Hagen. Tiled++: An enhanced tiled hi-res display wall. *IEEE Transactions on Visualization and Computer Graphics*, 16(1):120–132, 2010.
- [Fek04] Jean-Daniel Fekete. The infovis toolkit. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 167–174. IEEE, 2004.
- [Fit54] Paul M. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *J. Exp. Psych.*, 47:381–391, 1954.
- [FKGR09] Stephanie Foehrenbach, Werner A. König, Jens Gerken, and Harald Reiterer. Tactile feedback enhanced hand gesture interaction at large, high-resolution displays. *Journal of Visual Languages and Computing*, 20(5):341–351, 2009.
- [FKK07] Scott Frees, G. Drew Kessler, and Edwin Kay. Prism interaction for enhancing control in immersive virtual environments. *ACM Trans. Comput.-Hum. Interact.*, 14(1), 2007.
- [Fur86] George W. Furnas. Generalized Fisheye Views. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI)*, pages 16–23. ACM, 1986.
- [FVB06] Clifton Forlines, Daniel Vogel, and Ravin Balakrishnan. Hybridpointing: fluid switching between absolute and relative pointing with a direct input device. In *UIST '06: Proceedings of the 19th annual ACM symposium on User interface software and technology*, pages 211–220. ACM, 2006.
- [GB04] Yves Guiard and Michel Beaudouin-Lafon. Target acquisition in multiscale electronic worlds. *Int. J. Hum.-Comput. Stud.*, 61(6):875–905, 2004.
- [GGSF59] Eleanor J. Gibson, James J. Gibson, Olin W. Smith, and Howard Flock. Motion parallax as a determinant of perceived depth. *Journal of Experimental Psychology*, 58(1):40–51, 1959.
- [GKE⁺11] Tony Gjerlufsen, Clemens Nylandsted Klokmoose, James Eagan, Clément Pillias, and Michel Beaudouin-Lafon. Shared substance: developing flexible multi-surface applications. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, pages 3383–3392. ACM, 2011.
- [Goo11] Google. Liquid galaxy. <http://www.google.com/earth/explore/showcase/liquidgalaxy.html>, 2011. Last accessed 2012-03-02.
- [GP96] Thomas R.G. Green and Marian Petre. Usability analysis of visual programming environments: a ‘cognitive dimensions’ framework. *Journal of Visual Languages and Computing*, 7(2):131–174, 1996.
- [Gru01] Jonathan Grudin. Partitioning digital worlds: focal and peripheral awareness in multiple monitor use. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 458–465. ACM, 2001.
- [Gui87] Yves Guiard. Asymmetric division of labor in human skilled bimanual action: The kinematic chain as a model. *Journal of Motor Behavior*, 19:486–517, 1987.
- [Gut02] Carl Gutwin. Improving focus targeting in interactive fisheye views. In *CHI '02: Proc. Human Factors in Computing Systems*, pages 267–274. ACM, 2002.

- [HB11] Tom Heath and Christian Bizer. *Linked Data: Evolving the Web into a Global Data Space*. Morgan & Claypool, February 2011. 136 pages.
- [HBP02] Kasper Hornbæk, Benjamin B. Bederson, and Catherine Plaisant. Navigation patterns and usability of zoomable user interfaces with and without an overview. *ACM Trans. Comput.-Hum. Interact.*, 9(4):362–389, 2002.
- [HCL05] Jeffrey Heer, Stuart K. Card, and James A. Landay. prefuse: a toolkit for interactive information visualization. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '05, pages 421–430. ACM, 2005.
- [HDD⁺04] Stéphane Huot, Cédric Dumas, Pierre Dragicevic, Jean-Daniel Fekete, and Gérard Héron. The magglite post-wimp toolkit: draw it, connect it and run it. In *Proceedings of the 17th annual ACM symposium on User interface software and technology*, UIST '04, pages 257–266. ACM, 2004.
- [Hea08] Tom Heath. How will we interact with the web of data? *IEEE Internet Computing*, 12:88–91, 2008.
- [Hen12] James Hendler. The Semantic Web, It's for Real. <http://www.slideshare.net/jahendler/semantic-web-what-it-is-and-why-you-should-care>, 2012. Slide 14 – Last accessed 2012-03-01.
- [HF01] Kasper Hornbæk and Erik Frøkjær. Reading of electronic documents: the usability of linear, fisheye, and overview+detail interfaces. In *CHI '01: Proc. Human Factors in Computing Systems*, pages 293–300. ACM, 2001.
- [HFM07] Nathalie Henry, Jean-Daniel Fekete, and Michael J. McGuffin. Nodetrix: a hybrid visualization of social networks. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1302–1309, 2007.
- [HHN⁺02] Greg Humphreys, Mike Houston, Ren Ng, Randall Frank, Sean Ahern, Peter D. Kirchner, and James T. Klosowski. Chromium: a stream-processing framework for interactive rendering on clusters. *ACM Trans. Graph.*, 21(3):693–702, 2002.
- [HMK05] David Huynh, Stefano Mazzocchi, and David Karger. Piggy Bank: Experience the Semantic Web Inside Your Web Browser. In *Proc. of the 4th Int. Semantic Web Conf. (ISWC2005)*, LNCS 3729, pages 413–430, November 2005.
- [HMM00] Ivan Herman, Guy Melancon, and M. Scott Marshall. Graph visualization and navigation in information visualization: A survey. *IEEE Trans. on Visualization and Computer Graphics*, 06(1):24–43, 2000.
- [Hol06] Danny Holten. Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *IEEE Trans. on Visualization and Computer Graphics*, 12(5):741–748, 2006.
- [HPDR10] Chris Hundhausen, Emmanuel Pietriga, Paloma Díaz, and Mary Beth Rosson, editors. *Proceedings of the 2010 Symposium on Visual Languages and Human-Centric Computing*. IEEE Computer Society, September 2010. 276 pages.
- [HvOH06] Michiel Hildebrand, Jacco van Ossenbruggen, and Lynda Hardman. Ifacet: A Browser for Heterogeneous Semantic Web Repositories. In *Lecture Notes in Computer Science (LNCS 4273)*, *Proceedings of the 5th International Semantic Web Conference (ISWC 2006)*, pages 272–285. Springer, November 2006.

- [IBM12] IBM. Bringing big data to the enterprise. <http://www-01.ibm.com/software/data/bigdata/>, 2012. Last accessed 2012-02-23.
- [IH00] Takeo Igarashi and Ken Hinckley. Speed-dependent automatic zooming for browsing large documents. In *Proceedings of the 13th annual ACM symposium on User interface software and technology*, UIST '00, pages 139–148. ACM, 2000.
- [JGE12] Waqas Javed, Sohaib Ghani, and Niklas Elmqvist. PolyZoom: Multiscale and Multifocus Exploration in 2D Visual Spaces. In *CHI '12: Proceedings of the 30th international conference on Human factors in computing systems*. ACM, 2012. In press.
- [JGH⁺08] Robert J.K. Jacob, Audrey Girouard, Leanne M. Hirshfield, Michael S. Horn, Orit Shaer, Erin Treacy Solovey, and Jamie Zigelbaum. Reality-based interaction: a framework for post-WIMP interfaces. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08, pages 201–210. ACM, 2008.
- [JS92] Robert J.K. Jacob and Linda Sibert. The perceptual structure of multidimensional input device selection. In *CHI '92: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 211–218. ACM, 1992.
- [JW01] Oskar Juhlin and Alexandra Weilenmann. Decentralizing the control room: mobile work and institutional order. In *Proceedings of the seventh conference on European Conference on Computer Supported Cooperative Work*, pages 379–397. Kluwer Academic Publishers, 2001.
- [KB95] Paul Kabbash and William Buxton. The “prince” technique: Fitts’ law and selection using area cursors. In *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 273–279. ACM/Addison-Wesley, 1995.
- [KB09] Clemens Nylandsted Klokmoose and Michel Beaudouin-Lafon. Vigo: instrumental interaction in multi-surface environments. In *Proceedings of the 27th international conference on Human factors in computing systems*, CHI '09, pages 869–878. ACM, 2009.
- [KC04] Graham Klyne and Jeremy J. Carroll. Resource Description Framework (RDF): Concepts and Abstract Syntax, W3C Recommendation. <http://www.w3.org/TR/rdf-concepts/>, February 2004.
- [Kei02] Daniel A. Keim. Information Visualization and Visual Data Mining. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):1–8, January 2002.
- [KKPS03] José Kahan, Marja-Riitta Koivunen, Emmanuel Pietriga, and Ralph Swick. Cross-Application Data Interaction: Shared Bookmarks and IsaViz, 2003. WWW '03: the 12th World Wide Web Conference (Developer’s day).
- [KMSZ06] Daniel A. Keim, Florian Mansmann, Jörn Schneidewind, and Hartmut Ziegler. Challenges in Visual Data Analysis. In *Proceedings of the 10th International Conference on Information Visualization (IV)*, pages 9–16. IEEE, 2006.
- [Lec03] Eric Lecolinet. A molecular architecture for creating advanced guis. In *Proceedings of the 16th annual ACM symposium on User interface software and technology*, UIST '03, pages 135–144. ACM, 2003.
- [Lie94] Henry Lieberman. Powers of ten thousand: navigating in large information spaces. In *Proceedings of the 7th annual ACM symposium on User interface software and technology*, UIST '94, pages 15–16. ACM, 1994.

- [LPP⁺06] Bongshin Lee, Catherine Plaisant, Cynthia Sims Parr, Jean-Daniel Fekete, and Nathalie Henry. Task taxonomy for graph visualization. In *BELIV '06*, pages 1–5. ACM, 2006.
- [LZB98] Andrea Leganchuk, Shumin Zhai, and William Buxton. Manual and cognitive benefits of two-handed input: an experimental study. *ACM Trans. Comput.-Hum. Interact.*, 5(4):326–359, 1998.
- [Mac99] Wendy E. Mackay. Is paper safer? the role of paper flight strips in air traffic control. *ACM Trans. Comput.-Hum. Interact.*, 6:311–340, December 1999.
- [MBN⁺02] Brad A. Myers, Rishi Bhatnagar, Jeffrey Nichols, Choon Hong Peck, Dave Kong, Robert Miller, and A. Chris Long. Interacting at a distance: measuring the performance of laser pointers and other devices. In *CHI '02: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 33–40. ACM, 2002.
- [MCH⁺09] Tomer Moscovich, Fanny Chevalier, Nathalie Henry, Emmanuel Pietriga, and Jean-Daniel Fekete. Topology-Aware Navigation in Large Networks. In *CHI '09: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 2319–2328. ACM, 2009.
- [MG03] Paul Mutton and Jennifer Golbeck. Visualization of semantic metadata and ontologies. In *IV '03: Proc. of the Seventh Int. Conf. on Information Visualization*, pages 300–305. IEEE, 2003.
- [MH04a] Jock D. Mackinlay and Jeffrey Heer. Wideband displays: mitigating multiple monitor seams. In *CHI '04 extended abstracts on Human factors in computing systems*, pages 1521–1524. ACM, 2004.
- [MH04b] Tomer Moscovich and John F. Hughes. Navigating documents with the virtual scroll ring. In *UIST '04: Proceedings of the 17th annual ACM symposium on User interface software and technology*, pages 57–60. ACM, 2004.
- [MI09] David C. McCallum and Pourang Irani. ARC-Pad: Absolute+Relative Cursor Positioning for Large Displays with a Mobile Touchscreen. In *UIST '09: Proceedings of the 22nd annual ACM symposium on User interface software and technology*, pages 153–156. ACM, 2009.
- [MJ01] I. Scott MacKenzie and Shaidah Jusoh. An evaluation of two input devices for remote pointing. In *EHCI '01: Proceedings of the 8th IFIP International Conference on Engineering for Human-Computer Interaction*, pages 235–250. Springer, 2001.
- [MLG10] Sylvain Malacria, Eric Lecolinet, and Yves Guiard. Clutch-free panning and integrated pan-zoom control on touch-sensitive surfaces: the cyclostar approach. In *CHI '10: Proceedings of the 28th international conference on Human factors in computing systems*, pages 2615–2624. ACM, 2010.
- [MM04] Franck Manola and Eric Miller. RDF Primer, W3C Recommendation. <http://www.w3.org/TR/rdf-primer/>, February 2004.
- [MMKN08] Peter McLachlan, Tamara Munzner, Eleftherios Koutsofios, and Stephen North. LiveRAC: Interactive Visual Exploration of System Management Time-series Data. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems (CHI)*, pages 1483–1492. ACM, 2008.

- [MMM⁺97] Brad A. Myers, Richard G. McDaniel, Robert C. Miller, Alan S. Ferreny, Andrew Faulring, Bruce D. Kyle, Andrew Mickish, Alex Klimovitski, and Patrick Doane. The amulet environment: New models for effective user interface software development. *IEEE Trans. Softw. Eng.*, 23:347–365, June 1997.
- [MR92] Brad A. Myers and Mary Beth Rosson. Survey on user interface programming. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 195–202. ACM, 1992.
- [MRB05] Shahzad Malik, Abhishek Ranjan, and Ravin Balakrishnan. Interacting with large displays from a distance with vision-tracked multi-finger gestural input. In *UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology*, pages 43–52. ACM, 2005.
- [MvH04] Deborah L. McGuinness and Frank van Harmelen. OWL Web Ontology Language Overview, W3C Recommendation. <http://www.w3.org/TR/owl-features/>, February 2004.
- [NBM⁺06] Dmitry Nekrasovski, Adam Bodnar, Joanna McGrenere, François Guimbretière, and Tamara Munzner. An evaluation of pan & zoom and rubber sheet navigation with and without an overview. In *CHI '06: Proc. Human Factors in Computing Systems*, pages 11–20. ACM, 2006.
- [NC95] Laurence Nigay and Joëlle Coutaz. A generic platform for addressing the multimodal challenge. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '95, pages 98–105. ACM Press/Addison-Wesley Publishing Co., 1995.
- [NPBL11] Mathieu Nancel, Emmanuel Pietriga, and Michel Beaudouin Lafon. Precision Pointing for Ultra-High-Resolution Wall Displays. Research Report RR-7624, INRIA, 05 2011.
- [NS00] Chris North and Ben Shneiderman. Snap-together visualization: a user interface for coordinating visualizations via relational schemata. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '00, pages 128–135. ACM, 2000.
- [NSS⁺06] Tao Ni, Greg S. Schmidt, Oliver G. Staadt, Mark A. Livingston, Robert Ball, and Richard May. A Survey of Large High-Resolution Display Technologies, Techniques, and Applications. In *VR '06: Proceedings of the IEEE conference on Virtual Reality*, pages 223–236. IEEE, 2006.
- [NWP⁺11] Mathieu Nancel, Julie Wagner, Emmanuel Pietriga, Olivier Chapuis, and Wendy Mackay. Mid-air Pan-and-Zoom on Wall-sized Displays. In *CHI '11: Proceedings of the 29th international conference on Human factors in computing systems*, pages 177–186. ACM, 2011.
- [ODD06] Eyal Oren, Renaud Delbru, and Stefan Decker. Extending Faceted Navigation for RDF Data. In *Lecture Notes in Computer Science (LNCS 4273), Proceedings of the 5th International Semantic Web Conference (ISWC 2006)*, pages 559–572. Springer, November 2006.
- [ON01] Dan R. Olsen, Jr. and Travis Nielsen. Laser pointer interaction. In *CHI '01: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 17–22. ACM, 2001.
- [OS02] Ji-Young Oh and Wolfgang Stürzlinger. Laser pointers as collaborative pointing devices. In *Proc. Graphics Interface*, pages 141–150, 2002.

- [PA08] Emmanuel Pietriga and Caroline Appert. Sigma Lenses: Focus-Context Transitions Combining Space, Time and Translucence. In *Proceedings of ACM CHI 2008 Conference on Human Factors and Computing Systems*, pages 1343–1352. ACM Press, April 2008.
- [PAB07] Emmanuel Pietriga, Caroline Appert, and Michel Beaudouin-Lafon. Pointing and Beyond: an Operationalization and Preliminary Evaluation of Multi-scale Searching. In *Proceedings of ACM CHI 2007 Conference on Human Factors and Computing Systems*, pages 1215–1224. ACM Press, April 2007.
- [Pac10] Pacific Northwest National Laboratory. Precision information environments: Envisioning the future of emergency management. <http://precisioninformation.org/?p=32>, 2010. Last accessed 2012-03-02.
- [Pac11] Pacific Northwest National Laboratory. Gap assessment in the emergency response community. http://precisioninformation.org/wp-content/uploads/2011/03/PNNL_PIE_Gap_Report_20110318.pdf, February 2011. Last accessed 2012-03-02.
- [PBA10] Emmanuel Pietriga, Olivier Bau, and Caroline Appert. Representation-Independent In-Place Magnification with Sigma Lenses. *IEEE Transactions on Visualization and Computer Graphics*, 16(1):455–467, 2010.
- [PBKL06] Emmanuel Pietriga, Chris Bizer, David Karger, and Ryan Lee. Fresnel - A Browser-Independent Presentation Vocabulary for RDF. In *Lecture Notes in Computer Science (LNCS 4273), Proceedings of the 5th International Semantic Web Conference (ISWC 2006)*, pages 158–171. Springer, November 2006.
- [PCS95] Catherine Plaisant, David Carr, and Ben Shneiderman. Image-browser taxonomy and guidelines for designers. *IEEE Software*, 12(2):21–32, 1995.
- [PCS⁺12] Emmanuel Pietriga, Pierre Cubaud, Joseph Schwarz, Romain Primet, Marcus Schilling, Denis Barkats, Emilio Barrios, and Baltasar Vila Vilaro. Interaction Design Challenges and Solutions for ALMA Operations Monitoring and Control. In *Astronomical Telescopes and Instrumentation*. SPIE, 2012. To appear.
- [PD84] Thomas Porter and Tom Duff. Compositing digital images. In *SIGGRAPH '84: Proc. of the 11th Conf. on Computer Graphics and Interactive Techniques*, pages 253–259. ACM, 1984.
- [PF93] Ken Perlin and David Fox. Pad: an alternative approach to the computer interface. In *SIGGRAPH '93: Proc. of the Conf. on Computer Graphics and Interactive Techniques*, pages 57–64. ACM, 1993.
- [PHNP11] Emmanuel Pietriga, Stéphane Huot, Mathieu Nancel, and Romain Primet. Rapid Development of User Interfaces on Cluster-Driven Wall Displays with jBricks. In *EICS '11: Proceedings of the 2nd ACM SIGCHI symposium on Engineering interactive computing systems*, pages 185–190. ACM, 2011.
- [Pie02a] Emmanuel Pietriga. *Environnements et langages de programmation visuels pour le traitement de documents structurés*. PhD thesis, Institut National Polytechnique de Grenoble, November 2002. <http://tel.archives-ouvertes.fr/tel-00125472>.
- [Pie02b] Emmanuel Pietriga. IsaViz: a Visual Environment for Browsing and Authoring RDF Models, 2002. WWW '02: the 11th World Wide Web Conference (Developer's day).
- [Pie03] Emmanuel Pietriga. Styling RDF Graphs with GSS. *XML.com*, December 2003. <http://www.xml.com/pub/a/2003/12/03/gss.html>.

- [Pie05a] Emmanuel Pietriga. A Toolkit for Addressing HCI Issues in Visual Language Environments. In *IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC'05)*, pages 145–152, September 2005.
- [Pie05b] Emmanuel Pietriga. Fresnel Selector Language for RDF, November 2005. <http://www.w3.org/2005/04/fresnel-info/fsl/>.
- [Pie06] Emmanuel Pietriga. Semantic Web Data Visualization with Graph Style Sheets. In *ACM Symposium on Software Visualization (SoftVis'06)*, pages 177–178, September 2006.
- [PL09] Emmanuel Pietriga and Ryan Lee. Langages et outils pour la visualisation et la manipulation de données du Web sémantique. *Technique et Science Informatiques (TSI) Lavoisier*, 28(2):173–197, February 2009.
- [Pla04] Catherine Plaisant. The challenge of information visualization evaluation. In *AVI '04: Proc. of the working Conf. on Advanced visual interfaces*, pages 109–116. ACM, 2004.
- [PLVB00] Stuart Pook, Eric Lecolinet, Guy Vaysseix, and Emmanuel Barillot. Context and interaction in zoomable user interfaces. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '00, pages 227–231. ACM, 2000.
- [Pow11] Brian H. Powell. The keys to solving the world's top engineering challenges. In *IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, pages 4–4, September 2011.
- [PPCP12] Cyprien Pindat, Emmanuel Pietriga, Olivier Chapuis, and Claude Puech. JellyLens: Content-Aware Adaptive Lenses. In *UIST '12: Proceedings of the 25th Symposium on User Interface Software and Technology*. ACM, 2012. In press.
- [PS05] Eric Prud'hommeaux and Andy Seaborne. SPARQL Query Language for RDF. <http://www.w3.org/TR/rdf-sparql-query/>, April 2005.
- [PVD01] Emmanuel Pietriga and Jean-Yves Vion-Dury. VXT: Visual XML Transformer. In *IEEE Symposia on Human-Centric Computing (HCC)*, pages 404–405, September 2001.
- [PVDQ01] Emmanuel Pietriga, Jean-Yves Vion-Dury, and Vincent Quint. VXT: a visual approach to XML transformations. In *DocEng '01: Proceedings of the 2001 ACM Symposium on Document Engineering*, pages 1–10. ACM Press, 2001.
- [QHK03] Dennis Quan, David Huynh, and David R. Karger. Haystack: A Platform for Authoring End User Semantic Web Applications. In *Proc. of the 2nd Int. Semantic Web Conf. (ISWC2003)*, pages 738–753, October 2003.
- [QSA12] QSApp. Quicksilver – Mac OS X at your fingertips. <http://qsapp.com>, 2012. Last accessed 2012-03-01.
- [RCB⁺05] George G. Robertson, Mary Czerwinski, Patrick Baudisch, Brian Meyers, Daniel Robbins, Greg Smith, and Desney Tan. The large-display user experience. *IEEE CG&A*, 25:44–51, July 2005.
- [RCM93] George G. Robertson, Stuart K. Card, and Jack D. Mackinlay. Information visualization using 3d interactive animation. *Communication of the ACM*, 36(4):56–71, 1993.
- [RJH11] Mikkel Rønne Jakobsen and Kasper Hornbæk. Sizing up visualizations: effects of display size in focus+context, overview+detail, and zooming interfaces. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, pages 1451–1460. ACM, 2011.

- [SADK⁺09] Lauren Shupp, Christopher Andrews, Margaret Dickey-Kurdziolek, Beth Yost, and Chris North. Shaping the display of the future: The effects of display size and curvature on user performance and insights. *Human Computer Interaction*, 24:230–272, April 2009.
- [SB94] Manojit Sarkar and Marc H. Brown. Graphical fisheye views. *Communications of the ACM*, 37(12):73–83, 1994.
- [SBLH06] Nigel Shadbolt, Tim Berners-Lee, and Wendy Hall. The Semantic Web Revisited. *IEEE Intelligent Systems*, 21(3):96–101, 2006.
- [SHL03] Timothy A. Sandstrom, Chris Henze, and Creon Levit. The hyperwall. In *CMV '03: Proceedings of the conference on Coordinated and Multiple Views In Exploratory Visualization*, pages 124–133. IEEE, 2003.
- [Shn83] Ben Shneiderman. Direct manipulation: a step beyond programming languages. *IEEE Computer*, 16(8):57–69, 1983.
- [Shn96] Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *VL '96: Proc. of the 1996 IEEE Symp. on Visual Languages*, page 336. IEEE, 1996.
- [Sim05] Simile. Longwell RDF Browser, 2003-2005. <http://simile.mit.edu/longwell/>.
- [SJN08] Marcos Serrano, David Juras, and Laurence Nigay. A three-dimensional characterization space of software components for rapidly developing multimodal interfaces. In *Proceedings of the 10th international conference on Multimodal interfaces, ICMI '08*, pages 149–156. ACM, 2008.
- [sK06] m. c. schraefel and David Karger. The Pathetic Fallacy of RDF. In *Int. Workshop on the Semantic Web and User Interaction (SWUI)*, 2006.
- [SM04] R. William Soukoreff and I. Scott MacKenzie. Towards a standard for pointing device evaluation, perspectives on 27 years of fitts' law research in hci. *Int. J. Hum.-Comput. Stud.*, 61(6):751–789, 2004.
- [SPPS11] Marcus Schilling, Romain Primet, Emmanuel Pietriga, and Josph Schwarz. Human Computer Interaction in the ALMA Control Room. In *ADASS '11: Proceedings of the Astronomical Data Analysis Software and Systems*, pages 137–140, 2011.
- [SPSG10] Joseph Schwarz, Emmanuel Pietriga, Marcus Schilling, and Preben Grosbol. Goodbye to WIMPs: A Scalable Interface for ALMA Operations. *ADASS '10: Proceedings of the Astronomical Data Analysis Software and Systems*, 442:247–250, 2010.
- [sSO⁺05] m. c. schraefel, Daniel Smith, Alisdair Owens, Alistair Russell, and Craig Harris. The evolving mSpace platform: leveraging the Semantic Web on the Trail of the Memex. In *16th ACM Conference on Hypertext and Hypermedia*, pages 174–183, September 2005.
- [SSTR93] Manojit Sarkar, Scott S. Snibbe, Oren J. Tversky, and Steven P. Reiss. Stretching the rubber sheet: a metaphor for viewing large layouts on small screens. In *UIST '93: Proc. ACM Symp. on User Interface Software and Technology*, pages 81–91. ACM, 1993.
- [SVS⁺05] Reto Stockli, Eric Vermote, Nazmi Saleous, Robert Simmon, and David Herring. The Blue Marble Next Generation - A true color earth dataset including seasonal dynamics from MODIS. Published by the NASA Earth Observatory, 2005.

- [TC11] Benjamin Tissoires and Stéphane Conversy. Hayaku: designing and optimizing finely tuned and portable interactive graphics with a graphical compiler. In *Proceedings of the 3rd ACM SIGCHI symposium on Engineering interactive computing systems*, EICS '11, pages 117–126, New York, NY, USA, 2011. ACM.
- [TDO07] Giovanni Tummarello, Renaud Delbru, and Eyal Oren. Sindice. com: Weaving the open linked data. In *Proc. 6th Int'l Semantic Web Conf. and 2nd Asian Semantic Web Conf. (ISWC+ASWC 07) – LNCS 4825*, pages 552–565. Springer-Verlag, 2007.
- [TXZ⁺05] Kewei Tu, Miao Xiong, Lei Zhang, Haiping Zhu, Jie Zhang, and Yong Yu. Towards imaging large-scale ontologies for quick understanding and analysis. In *Proc. of the 4th Int. Semantic Web Conf. (ISWC2005)*, LNCS 3729, pages 702–715. Springer-Verlag, 2005.
- [VB05] Daniel Vogel and Ravin Balakrishnan. Distant freehand pointing and clicking on very large, high resolution displays. In *UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology*, pages 33–42. ACM, 2005.
- [VDLP02] Jean-Yves Vion-Dury, Veronika Lux, and Emmanuel Pietriga. Experimenting with the circus language for XML modeling and transformation. In *DocEng '02: Proceedings of the 2002 ACM symposium on Document engineering*, pages 82–87. ACM Press, 2002.
- [VDP01] Jean-Yves Vion-Dury and Emmanuel Pietriga. A Formal Study of a Visual Language for the Visualization of Document Type Definition. In *IEEE Symposia on Human-Centric Computing (HCC)*, pages 52–59, September 2001.
- [VPJM05] Diogo F. Veiga, Pedro de Stege Cecconello, José Eduard de Lucca, and Luismar Marques Porto. Extension of the IsaViz Software for the Representation of Metabolic and Regulatory Networks. *Brazilian Archives of Biology and Technology*, 48:197–205, 2005.
- [W3C98] W3C. Cascading Style Sheets, level 2, May 1998. <http://www.w3.org/TR/1998/REC-CSS2-19980512>.
- [W3C99] W3C. XSL Transformations (XSLT), November 1999. <http://www.w3.org/TR/xslt>.
- [W3C03] W3C. Wine Ontology, 2003. <http://www.w3.org/2005/04/fresnel-info/fsl/>.
- [W3C07] W3C. Scalable Vector Graphics (SVG) - XML Graphics for the Web. <http://www.w3.org/Graphics/SVG/>, February 2007.
- [War04] Colin Ware. *Information visualization: perception for design (2nd edition)*. Morgan Kaufmann, 2004. 486 pages.
- [Whe03] Elaine Wherry. Scroll ring performance evaluation. In *CHI '03: CHI '03 extended abstracts on Human factors in computing systems*, pages 758–759. ACM, 2003.
- [WL95] Colin Ware and Marlon Lewis. The DragMag image magnifier. In *CHI '95 conference companion, Human Factors in Computing Systems*, pages 407–408. ACM Press, 1995.
- [WP06] Taowei David Wang and Bijan Parsia. CropCircles: Topology Sensitive Visualization of OWL Class Hierarchies. In *Lecture Notes in Computer Science (LNCS 4273), Proceedings of the 5th International Semantic Web Conference (ISWC 2006)*, pages 695–708. Springer, November 2006.
- [Wur89] Richard S. Wurman. *Information Anxiety*. Doubleday, 1989. 350 pages.

- [YHN07] Beth Yost, Yonca Hacıahmetoglu, and Chris North. Beyond visual acuity: the perceptual scalability of information visualizations for large displays. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 101–110. ACM, 2007.
- [ZBL⁺10] Jamie Zigelbaum, Alan Browning, Daniel Leithinger, Olivier Bau, and Hiroshi Ishii. g-stalt: a chirocentric, spatiotemporal, and telekinetic gestural interface. In *Proceedings of the fourth international conference on Tangible, embedded, and embodied interaction*, TEI '10, pages 261–264. ACM, 2010.
- [ZCPB11] Jian Zhao, Fanny Chevalier, Emmanuel Pietriga, and Ravin Balakrishnan. Exploratory Analysis of Time-series with ChronoLenses. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2422–2431, 2011.
- [ZKB02] Jürgen Ziegler, Christoph Kunz, and Veit Botsch. Matrix browser: visualizing and exploring large networked information spaces. In *CHI '02 extended abstracts*, pages 602–603. ACM, 2002.
- [ZMB96] Shumin Zhai, Paul Milgram, and William Buxton. The influence of muscle groups on performance of multiple degree-of-freedom input. In *CHI '96: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 308–315. ACM, 1996.
- [ZR08] Nan Zang and Mary Beth Rosson. What's in a mashup? and why? studying the perceptions of web-active end users. In *Proceedings of the 2008 IEEE Symposium on Visual Languages and Human-Centric Computing*, VLHCC '08, pages 31–38. IEEE Computer Society, 2008.

Appendices

Publications

The following pages contain nine publications that give an overview of my work since I joined the In-Situ project-team at INRIA Saclay – Île-de-France in October 2004.

Figure A.1 puts those publications in context.

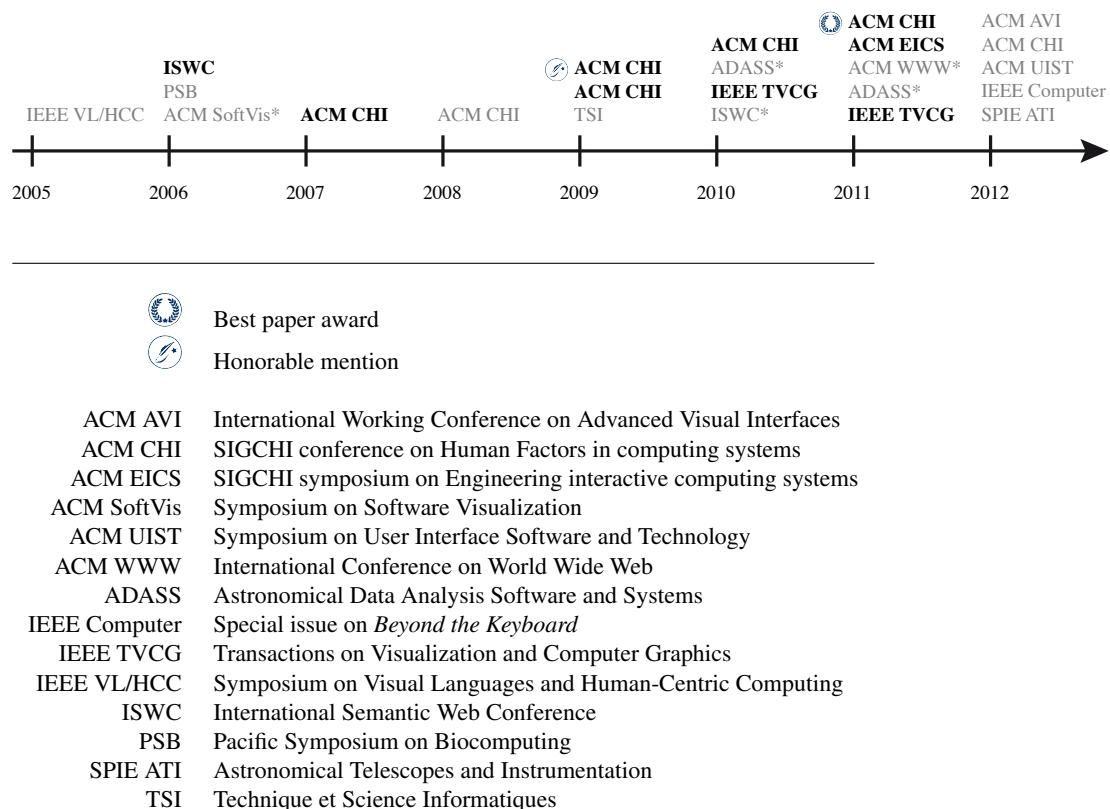


Figure A.1 : Selected publications (in bold) are available in this appendix. Other publications can be downloaded from the publisher's Digital Library or the HAL-INRIA open archive: <http://hal.inria.fr>. Publications marked with a * are short papers (refereed, and archived in proceedings).

Selected Publications (2006-2011)

- [1] **Emmanuel Pietriga**, Chris Bizer, David Karger, and Ryan Lee. Fresnel - A Browser-Independent Presentation Vocabulary for RDF. In *Lecture Notes in Computer Science (LNCS 4273), Proceedings of the 5th International Semantic Web Conference (ISWC 2006)*, pages 158–171. Springer, 2006.
- [2] **Emmanuel Pietriga**, Caroline Appert, and Michel Beaudouin-Lafon. Pointing and Beyond: an Operationalization and Preliminary Evaluation of Multi-scale Searching. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 1215–1224. ACM, 2007.
- [3] Olivier Chapuis, Jean-Baptiste Labrune, and **Emmanuel Pietriga**. DynaSpot: Speed-Dependent Area Cursor. In *CHI '09: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 1391–1400. ACM, 2009. .
- [4] Tomer Moscovich, Fanny Chevalier, Nathalie Henry, **Emmanuel Pietriga**, and Jean-Daniel Fekete. Topology-Aware Navigation in Large Networks. In *CHI '09: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 2319–2328. ACM, 2009.
- [5] **Emmanuel Pietriga**, Olivier Bau, and Caroline Appert. Representation-Independent In-Place Magnification with Sigma Lenses. *IEEE Transactions on Visualization and Computer Graphics*, 16(1):455–467, 2010.
- [6] Caroline Appert, Olivier Chapuis, and **Emmanuel Pietriga**. High-precision Magnification Lenses. In *CHI '10: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 273–282, 2010.
- [7] Jian Zhao, Fanny Chevalier, **Emmanuel Pietriga**, and Ravin Balakrishnan. Exploratory Analysis of Time-series with ChronoLenses. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2422–2431, 2011.
- [8] Mathieu Nancel, Julie Wagner, **Emmanuel Pietriga**, Olivier Chapuis, and Wendy Mackay. Mid-air Pan-and-Zoom on Wall-sized Displays. In *CHI '11: Proceedings of the 29th international conference on Human factors in computing systems*, pages 177–186. ACM, 2011. .
- [9] **Emmanuel Pietriga**, Stéphane Huot, Mathieu Nancel, and Romain Primet. Rapid Development of User Interfaces on Cluster-Driven Wall Displays with jBricks. In *EICS '11: Proceedings of the 2nd ACM SIGCHI symposium on Engineering interactive computing systems*, pages 185–190. ACM, 2011.

Fresnel: A Browser-Independent Presentation Vocabulary for RDF

Emmanuel Pietriga¹, Christian Bizer², David Karger³, and Ryan Lee^{3,4}

¹ INRIA & Laboratoire de Recherche en Informatique (LRI), Orsay, France

`emmanuel.pietriga@inria.fr`

² Freie Universität Berlin, Germany

`chris@bizer.de`

³ MIT CSAIL, Cambridge, MA, USA

`karger@mit.edu`

⁴ W3C (World Wide Web Consortium), Cambridge, MA, USA

`ryanlee@w3.org`

Abstract. Semantic Web browsers and other tools aimed at displaying RDF data to end users are all concerned with the same problem: presenting content primarily intended for machine consumption in a human-readable way. Their solutions differ but in the end address the same two high-level issues, no matter the underlying representation paradigm: specifying (i) *what* information contained in RDF models should be presented (content selection) and (ii) *how* this information should be presented (content formatting and styling). However, each tool currently relies on its own *ad hoc* mechanisms and vocabulary for specifying RDF presentation knowledge, making it difficult to share and reuse such knowledge across applications. Recognizing the general need for presenting RDF content to users and wanting to promote the exchange of presentation knowledge, we designed Fresnel as a browser-independent vocabulary of core RDF display concepts. In this paper we describe Fresnel's main concepts and present several RDF browsers and visualization tools that have adopted the vocabulary so far.

1 Introduction

RDF (Resource Description Framework) is designed to facilitate machine interpretability of information and does not define a visual presentation model since human readability is not one of its stated goals. Displaying RDF data in a user-friendly manner is a problem addressed by various types of applications using different representation paradigms. Web-based tools such as Longwell [1] (see Figure 1-a) and Piggy-Bank [2] use nested box layouts, or table-like layouts (e.g. Brownsauce [3], Noadster [4], Swoop [5]) for displaying properties of RDF resources with varying levels of details. Other tools like IsaViz [6] (see Figure 1-b) and Welkin [7] represent RDF models as node-link diagrams, explicitly showing their graph structure. A third approach combines these paradigms and extends them with specialized user interface widgets designed for specific information items like calendar data, tree structures, or even DNA sequences, providing advanced navigation tools and other interaction capabilities: Haystack [8] (see Figure 1-c), mSpace [9] and Tabulator [10].

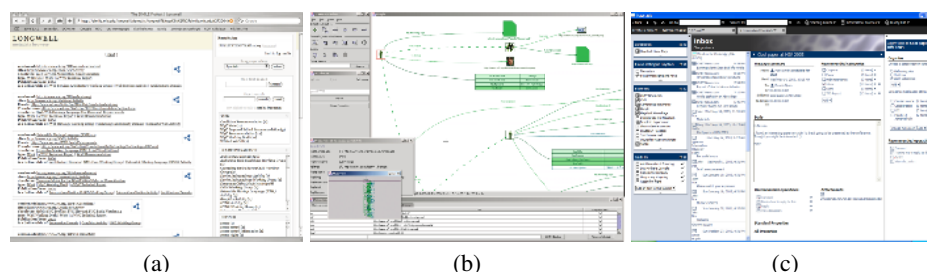


Fig. 1. Various types of RDF browsers: Longwell, IsaViz and Haystack

Such applications are confronted with the same two issues, independently of the underlying representation paradigm and interface capabilities: selecting what content to show and specifying how to format and style this content. Each application takes its own approach and defines its own vocabulary to specify how to present data to users. As with other kinds of knowledge, we believe that being able to share what we consider *presentation knowledge* makes sense in the context of the Semantic Web and that being able to exchange and reuse presentation knowledge between browsers and other visualization tools will benefit both programmers and end users. However, the current diversity of approaches and vocabularies for representing this knowledge makes such exchange and reuse difficult at best, if not impossible.

1.1 Related Work

Early RDF visualization tools rendered RDF models in a predefined, non-customizable way [3]. Recent tools provide more flexible visualizations that can be customized by writing style sheets, transformations, or templates, following either a declarative or a procedural approach.

Procedural approaches consider the presentation process as a series of transformation steps. One such approach consists in using XSLT to transform RDF graphs encoded as RDF/XML trees in an environment such as Cocoon [11]. Authoring XSLT templates and XPath expressions to handle arbitrary RDF/XML is complex, if not impossible, considering the many potential serializations of a given RDF graph and the present lack of a commonly accepted RDF canonicalization in XML [12]. This problem has been partly addressed by Xenon [13], an RDF style sheet ontology that builds on the ideas of XSLT but combines a recursive template mechanism with SPARQL as an RDF-specific selector language. Xenon succeeds in addressing XSLT's RDF canonicalization problem but still has a drawback common to all procedural approaches, that transformation rules are tied to a specific display paradigm and output format, thus preventing the reuse of presentation knowledge across applications.

Declarative approaches are based on formatting and styling rules applied to a generic representation of the content. They can be compared to XHTML+CSS, which has been successful for the classic Web. The Haystack Slide ontology [14], used to describe how Haystack display widgets are laid out, is one example. Another is IsaViz's Graph Style Sheets [15], which modifies the formatting, styling, and visibility of RDF graph

160 E. Pietriga et al.

elements represented as node-link diagrams. The main drawback of the declarative approaches developed so far is that they make strong assumptions about, and are thus tied to, the specific display paradigm for which they have been developed and are therefore unlikely to be meaningful across different representation paradigms.

1.2 Toward the Specification of Presentation Knowledge

Providing a single global view of all the information contained in an RDF model is often not useful. The mass of data makes it difficult to extract information relevant to the current task and represents a significant cognitive overload for the user. From an abstract perspective, the first step of the presentation process thus consists in restricting the visualization to small but cohesive parts of the RDF graph, similarly to views in the database world. But identifying what content to show is not sufficient for making a human-friendly presentation from the information. To achieve this goal, the selected content items must be laid out properly and rendered with graphical attributes that favor legibility in order to facilitate general understanding of the displayed information. Relying solely on the content's structure and exploiting knowledge contained in the schema associated with the data is insufficient for producing sophisticated presentations and visualizations. The second step thus consists in formatting and styling selected content items.

Fresnel's goal is to provide an RDF vocabulary to encode information about how to present Semantic Web content to users (i.e., *what* content to show, and *how* to show it) as presentation knowledge that can be exchanged and reused between browsers and other visualization tools. However, we do not expect all applications, which do not necessarily rely on the same representation paradigms and formats, to exchange and reuse all formatting and styling instructions as some might not be appropriate for all paradigms. We therefore identified a set of core presentation concepts that are applicable across applications and which form the core modules of Fresnel. One of the design goals of these modules was to make them easy to learn and use, but also easy to implement in order to promote their adoption by many applications. On top of these modules, we have also begun to define additional Fresnel vocabulary items which are grouped in extension modules. The remainder of this article mainly focuses on the core selection and formatting modules. More information about extension modules can be found in the Fresnel User Manual [16].

2 Core Vocabulary Overview

Fresnel is an RDF vocabulary, described by an OWL ontology [16]. Fresnel presentation knowledge is thus expressed declaratively in RDF and relies on two foundational concepts: *lenses* and *formats* (see Figure 2). Lenses specify which properties of RDF resources are shown and how these properties are ordered while formats indicate how to format content selected by lenses and optionally generate additional static content and hooks in the form of CSS class names that can be used to style the output through external CSS style sheets. The following sections introduce the main vocabulary elements using the examples in Figures 3 and 4.

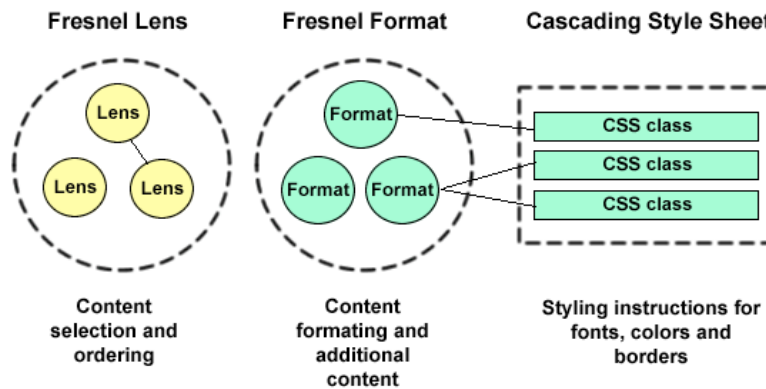


Fig. 2. Fresnel foundational concepts

Figure 3 shows a simple lens and associated formats used to present information about a person described with the FOAF vocabulary [17]. This figure also shows a possible rendering of such a resource, that a browser like Horus [18] or Longwell [1] could produce. Examples use the Notation 3 syntax [19].

2.1 Content Selection

The domain of a lens indicates the set of resources to which a lens applies (line 301: the lens applies to instances of class `foaf:Person`). Property `fresnel:showProperties` is used to specify what properties of these resources to show and in what order (lines 302-308). In this example, the values of both `fresnel:classLensDomain` and `fresnel:showProperties` are basic selectors, which take the form of plain URIs (represented here as qualified names), respectively identifying the class of resources and property types to select. More advanced selection expressions can be written using either FSL or SPARQL. They make it possible to associate lenses with untyped RDF resources, which do occur in real-world models since `rdf:type` properties are not mandatory. They can also be used to specify that a lens should display all properties of a given namespace, or any other complex selection condition(s) that can be represented by an FSL or SPARQL expression (see Section 3).

Fresnel Core provides additional constructs for specifying what properties of resources to display. The special value `fresnel:allProperties` is used when the list of properties that can potentially be associated with resources handled by a lens is unknown to the lens' author but should nevertheless be displayed. When it appears as a member of the list of properties to be shown by a lens, `fresnel:allProperties` designates the set of properties that are not explicitly designated by other property URI references in the list, except for properties that appear in the list of properties to hide (`fresnel:hideProperties`). Two other constructs are used to handle the potential irregularity of RDF data stemming from the fact that different authors might use similar terms coming from different vocabularies to make equivalent statements. Sets of such similar properties can be said to be `fresnel:alternateProperties`. For instance, `foaf:depiction`, `foaf:img` and `p3p:image` could be considered as providing

162 E. Pietriga et al.

```

(300) :PersonLens a fresnel:Lens ;
(301) fresnel:classLensDomain foaf:Person ;
(302) fresnel:showProperties (
(303)   foaf:name
(304)   foaf:mbox
(305)   [rdf:type fresnel:PropertyDescription;
(306)     fresnel:alternateProperties (
(307)       foaf:depiction foaf:img p3p:image )
(308)   ] ) .

(309) :nameFormat a fresnel:Format ;
(310)   fresnel:propertyFormatDomain foaf:name ;
(311)   fresnel:label "Name" .

(312) :mboxFormat a fresnel:Format ;
(313)   fresnel:propertyFormatDomain foaf:mbox ;
(314)   fresnel:label "Mailbox" ;
(315)   fresnel:value fresnel:externalLink ;
(316)   fresnel:valueFormat [ fresnel:contentAfter ", " ] .

(317) :depictFormat a fresnel:Format ;
(318)   fresnel:propertyFormatDomain foaf:depiction ;
(319)   fresnel:label fresnel:none ;
(320)   fresnel:value fresnel:image .

```



Fig. 3. A lens and some formats for presenting instances of class `foaf:Person`

the same information about resources displayed by a given lens. A browser using this lens would try to display the resource's `foaf:depiction`. If the latter did not exist, the browser would then look for `foaf:img` or `p3p:image` (see lines 305-307). Such knowledge can also be represented through ontology mapping mechanisms, but Fresnel provides this alternative as the ontology layer should not be made a requirement of the Fresnel presentation process. The other construct, `fresnel:mergeProperties`, is used to merge the values of related properties (e.g. `foaf:homepage` and `foaf:work-Homepage`) into one single set of values that can later be formatted as a whole.

The presentation of property values is not limited to a single level, and (possibly recursive) calls to lenses can be made to display details about the value of a property. Lenses used in this context are referred to as *sublenses*. Modifying the example of Figure 3, we specify in Figure 4 that the browser should render values of the property `foaf:knows` (lines 405-407) using another lens (`PersonLabelLens`, lines 410-413). The FSL expression (see Section 3) on line 406 specifies in an XPath-like manner that only values of `foaf:knows` that are instances of `foaf:Person` should be selected.

The sublens mechanism implies that a lens can recursively call itself as a sublens for displaying property values. In order to prevent infinite loops caused by such recursive calls, Fresnel defines a closure mechanism that allows Fresnel presentation designers to specify the maximum depth of the recursion.

2.2 Content Formatting

The default layout of selected information items is highly dependent on the browser's representation paradigm (e.g. nested box layout, node-link diagrams, etc.), but the final rendering can be customized by associating formatting and styling instructions with elements of the representation.

```
(400) :PersonLens a fresnel:Lens ;
(401)   fresnel:classLensDomain foaf:Person ;
(402)   fresnel:showProperties (
(403)     foaf:name
(404)     foaf:mbox
(405)     [rdf:type fresnel:PropertyDescription ;
(406)       fresnel:property "foaf:knows[foaf:Person]"^^fresnel:fslSelector;
(407)       fresnel:sublens :PersonLabelLens]
(408)   ) ;
(409)   fresnel:group :FOAFmainGroup .

(410) :PersonLabelLens a fresnel:Lens ;
(411)   fresnel:classLensDomain foaf:Person ;
(412)   fresnel:showProperties ( foaf:name ) ;
(413)   fresnel:group :FOAFsubGroup .

(414) :nameFormat a fresnel:Format ;
(415)   fresnel:propertyFormatDomain foaf:name ;
(416)   fresnel:label "Name" ;
(417)   fresnel:group :FOAFmainGroup .

(418) :mboxFormat a fresnel:Format ;
(419)   fresnel:propertyFormatDomain foaf:mbox ;
(420)   fresnel:label "Mailbox" ;
(421)   fresnel:value fresnel:externalLink ;
(422)   fresnel:valueFormat [ fresnel:contentAfter "," ] ;
(423)   fresnel:group :FOAFmainGroup .

(424) :friendsFormat a fresnel:Format ;
(425)   fresnel:propertyFormatDomain foaf:name ;
(426)   fresnel:label "Friends" ;
(427)   fresnel:group :FOAFsubGroup .

(428) :FOAFmainGroup a fresnel:Group .
(429) :FOAFsubGroup a fresnel:Group .
```

Name	Chris Bizer
Mailbox	chris@bizer.de , bizer@gmx.de
Friends	Emmanuel Pietriga Ryan Lee David Karger Stefano Mazzocchi

Fig. 4. An example of a lens using another lens to display some property values

Formats apply to resources, or to properties and their values, depending on the specified domain. The three example formats of Figure 3 apply respectively to the properties `foaf:name`, `foaf:mbox` and `foaf:depiction` (lines 310, 313, 318). Formats can be used to set properties' labels (lines 311, 314, 319). Property `fresnel:label` does not specify a particular layout but simply gives a text string that can be used to identify the property. Labels might already be defined for many properties (e.g., in the associated

164 E. Pietriga et al.

vocabulary description using `rdfs:label`), but such labels are not guaranteed to exist. Moreover, a given label might not always be the most appropriate depending on the context in which the property is displayed. For instance, the default label associated with property `foaf:name` in the FOAF schema is *name*. When displaying the persons known by the current person in Figure 4, this default label is replaced by *Friends* (line 426) so as to indicate the appropriate interpretation of the corresponding `foaf:name` property values in this context. The customization of labels also proves useful when displaying property values that are not direct properties of the current resource, as is made possible by the use of SPARQL or FSL expressions such as:

```
foaf:knows/*[airport:iataCode/text() = 'CDG']/foaf:name
```

which would require an explanatory label such as *Friends that leave near Paris*.

Formats can also give instructions regarding how to render values. For instance, line 315 indicates that `foaf:mbox` values should be rendered as clickable links (email addresses). Values of `foaf:depiction` should be fetched from the Web and rendered as bitmap images (line 320).

Property values can be grouped, and additional content such as commas and an ending period can be specified to present multi-valued properties (line 316: inserting a comma in-between each email address). CSS class names can also be associated with the various elements being formatted. These names appear in the output document and can be used to style the output by authoring and referencing CSS style sheets that use rules with the same class names as selectors.

2.3 Lens and Format Grouping

Lenses and formats can be associated through `fresnel:Groups` so that browsers can determine which lenses and formats work together. Fresnel groups are taken into account by browsers when selecting what format(s) to apply to the data selected by a given lens, as several formats might be applicable to the same property values.

Figure 4 illustrates the use of Fresnel groups to display different labels for the `foaf:name` property depending on the context in which the property is shown: the property is labeled *Name* when displayed in the context of the `PersonLens` lens, but is labeled *Friends* when displayed in the context of the `PersonLabelLens` lens. This is achieved by associating the `PersonLens` (lines 400-409) and the `nameFormat` (lines 414-417) to one group: `FOAFmainGroup`, and by associating the `PersonLabelLens` (lines 410-413) and the `friendsFormat` (lines 424-427) to a second group: `FOAFsubGroup`.

A Fresnel group can also serve as a placeholder for formatting instructions that apply to all formats associated with that group, thus making it possible to factorize the declarations. It is also typically used to declare group-wide data, relevant to both lenses and formats, such as namespace prefix bindings.

3 Fresnel Selectors

Selection in Fresnel occurs when specifying the domain of a lens or format and when specifying what properties of a resource a lens should show. Such selection expressions

identify elements of the RDF model to be presented; in other words, specific nodes and arcs in the graph. As we expect selection conditions to be of varying complexity, we allow them to be expressed using different languages in an attempt to balance expressive power against ease of use.

3.1 Basic Selectors

The simplest selectors, called basic selectors, take the form of plain URI references as shown in section 2. Depending on whether they are used as values of `fresnel:instanceLensDomain` or `fresnel:classLensDomain`, these URI references are interpreted respectively either as:

- URI equality constraints (the resource to be selected should be identified by this URI),
- or type constraints (the resources to be selected should be instances of the class identified by this URI).

Basic selectors are also used to identify properties, which are used for instance as values of `fresnel:showProperties` or `fresnel:alternateProperties`.

Basic selectors are easy to use but have very limited expressive power. For instance, they cannot be used to specify that a lens should apply to all instances of class `foaf:Person` that are the subject of at least five `foaf:knows` statements. More powerful languages are required to express such selection constraints.

3.2 Languages for Complex Selection Expressions

Fresnel presentation designers can use two different languages for expressing complex selection expressions. The first option is the SPARQL query language for RDF [20]. In the context of Fresnel, SPARQL queries must always return exactly one result set, meaning that only one variable is allowed in the query's SELECT clause. Figure 5-a gives an example of a lens whose domain is defined by a SPARQL expression. Alternatively, designers who prefer a more XPath-like approach, which proved to be a well-adapted selector language for XSLT, can use the Fresnel Selector Language (FSL). FSL is a language for modeling traversal paths in RDF graphs, designed to address the specific requirements of a selector language for Fresnel. It does not pretend to be a full so-called RDFPath language (contrary to XPR [21], an extension of FSL) but tries to be as simple as possible, both from usability and implementation perspectives. FSL is strongly inspired by XPath [22], reusing many of its concepts and syntactic constructs while adapting them to RDF's graph-based data model. RDF models are considered directed labeled graphs according to RDF Concepts and Abstract Syntax [23]. FSL is therefore fully independent from any serialization. A lens definition using two FSL expressions is shown in Figure 5-b. More information about FSL, including its grammar, data model and semantics is available in the FSL specification [24].

Applications implementing Fresnel are required to support basic selectors, and we expect a reasonable share of them to support the two other languages: SPARQL is gaining

166 E. Pietriga et al.

```
# (a) Lens for John Doe's mailboxes      (SPARQL)
:PersonLens a fresnel:Lens ;
    fresnel:instanceLensDomain
        "SELECT ?mbox WHERE ( ?x foaf:name 'John Doe' )
        ( ?x foaf:mbox ?mbox )"^^fresnel:sparqlSelector .

# (b) Lens for foaf:Person instances that know at least five other resources (FSL)
:PersonLens a fresnel:Lens ;
    fresnel:instanceLensDomain
        "foaf:Person[count(foaf:knows) >= 5]"^^fresnel:fslSelector ;
# and which shows the foaf:name property of all foaf:Person
# instances known by the current resource.
    fresnel:showProperties (
        "foaf:knows/foaf:Person/foaf:name"^^fresnel:fslSelector) .
```

Fig. 5. Examples of SPARQL and FSL expressions used in Fresnel lens definitions

momentum as a W3C recommendation, and four open-source Java implementations of FSL are already available¹ for HP's Jena Semantic Web Toolkit², for IsaViz (providing a visual FSL debugger) and for different versions of the Sesame RDF database³.

4 Implementations

Fresnel has been designed as an application- and output format-independent RDF presentation vocabulary. In this section we give an overview of various applications implementing Fresnel: Longwell [1] and Horus [18] which both render RDF data as HTML Web pages using nested box layouts, IsaViz [6] which represents RDF graphs as node-link diagrams, and Cardovan, a browser and lens editor based on the SWT GUI toolkit.

Longwell is a Web-based RDF browser whose foundational navigation paradigm is faceted browsing. Faceted browsing displays only the properties that are configured to be 'facets' (i.e., to be important for the user browsing data in one or more specific domains) using values for those fields as a means for zooming into a collection by selecting those items with a particular field-value pair.

The latest version of Longwell relies on the SIMILE Fresnel rendering engine, a Java library built on the Sesame triple store. The engine implements all of the Fresnel core vocabulary and the portion of the extended vocabulary relating to linking groups to CSS stylesheets as well as the option of using FSL as a selector language. The Fresnel engine output consists solely of an XML representation of the Fresnel lenses and formats as they apply to one resource. Longwell then applies an XSLT transformation to the XML to generate XHTML. The default XSLT stylesheet shipped with Longwell will generate a traditional nested box layout, as Horus does, but the stylesheet can be modified by XSLT developers to change the model as they see fit.

The left side of Figure 6 shows the rendering of a `foaf:Organization` resource using a lens that gives some details about the organization and lists its constituent members, all `foaf:Persons`, each listed with their corresponding nickname information to assist in identification.

¹ <http://dev.w3.org/cvsweb/java/classes/org/w3c/IsaViz/fresnel/>

² <http://jena.sourceforge.net>

³ <http://openrdf.org>

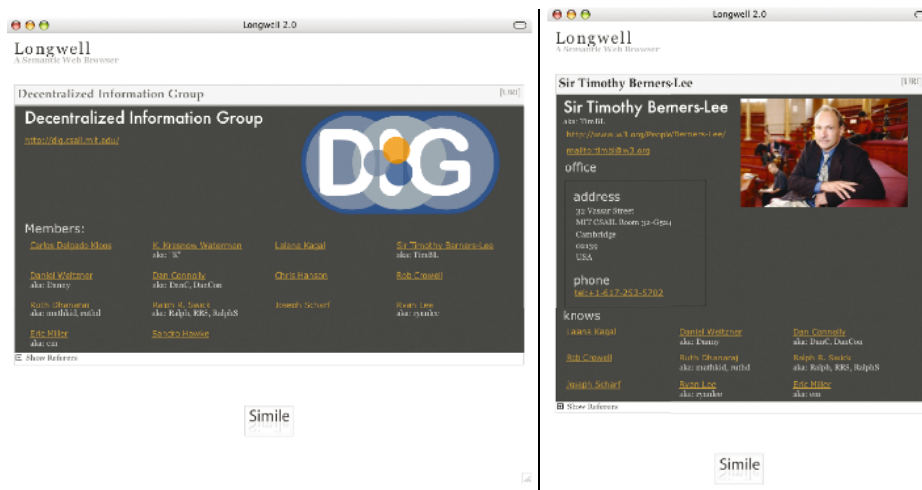


Fig. 6. Displaying a view of an organization (left) and a constituent member (right) in Longwell

The nickname list for each person is preceded by the string 'aka: ', added to the display by using the `fresnel:contentFirst` directive. The list is also comma separated, accomplished by setting `fresnel:contentAfter` to a comma. Clicking on a URI in the display brings the user to that URI; clicking on a textual label changes Longwell's focus to the resource represented by that label.

On the right side of Figure 6, the focus is on one specific member of the organization featured in the left side. A sublens is used to generate office contact details, and the same sublens used in the organization focus (left image) to describe an organization's members is used in the person focus (right image) to describe who this person claims to know.

Horus is an RDF browser that displays RDF information using a nested box layout. The browser provides a simple navigation paradigm for selecting RDF resources and allows users to switch between different lenses for rendering the resources. Horus supports Fresnel lenses and formats, which can be associated together using Fresnel groups. Groups can refer to external CSS style sheets which are used to define fonts, colors and borders. Horus supports basic selectors, but does not offer SPARQL and FSL as selector languages. Horus is implemented using PHP and is backed by a MySQL database. Applying a lens to an RDF resource results in an intermediate tree, which is formatted afterwards using the formats that are associated to the group of the selected lens. The ordered and formatted intermediate tree is then serialized into XHTML.

Figure 7 shows two different views on the same person in Horus. The view on the left uses a lens that displays many details about persons. The sentence *"This person knows the following people"* is a custom label for property `foaf:knows`. The disclaimer *"That a person knows somebody does..."* is static content added using property `fresnel:contentLast`. Some of the links are formatted as external links (`fresnel:value` formatting instruction set to `fresnel:externalLink`), while others refer to RDF resources in the knowledge base, and thus have a different rendering.

168 E. Pietriga et al.

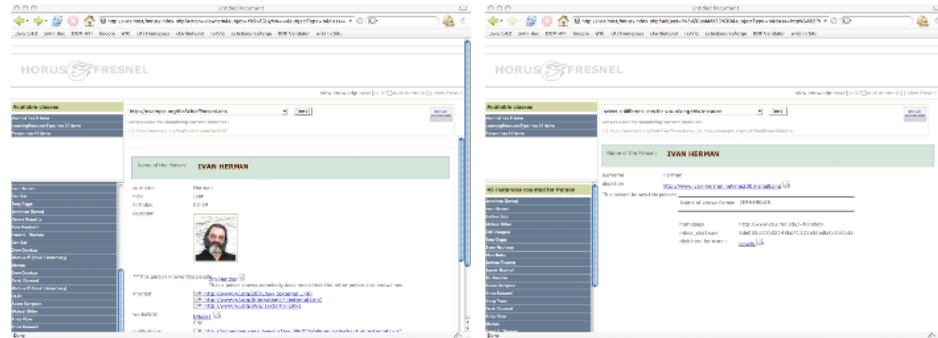


Fig. 7. Two different views on the same person in Horus: detailed view (left), friends view (right)

On the right side of Figure 7, the same person is shown using a different lens. This lens displays less details about the person itself, but refers to a second lens (used as a sub-lens) for displaying details about other persons known by this person. As the sub-lens belongs to a different group, another CSS class is used to style the names of the person's friends.

IsaViz is an RDF authoring environment representing RDF models as node-link diagrams. The interpretation of Fresnel in IsaViz is inspired by both Generalized Fisheye Views [25] and Magic Lenses [26]. Fresnel lenses, in conjunction with the formats associated with them through groups, are considered as “genuine” lenses that modify the visual appearance of objects below them.

Figure 8 (left) shows the default rendering of a region of an RDF model containing a `foaf:Person` resource. At this level of magnification, only a few of the many property values associated with the resource are visible. Users need to navigate in the graph in order to get to the values of properties, which can be cumbersome. Alternatively, users can select a Fresnel lens from the list of available lenses loaded in IsaViz through the graphical user interface. The selected lens is then tied to the mouse cursor, and when the lens hovers over a resource that matches its domain, the resource's visual appearance gets modified according to the lens and associated format(s). Resources that match the selected lens' domain are made visually prominent by rendering all other nodes and all arcs using shades of gray with minimum contrast. When the lens hovers over a resource, properties selected by the lens are temporarily rendered with highly-contrasted vivid colors and brought within the current view, closer to the main resource and reordered clockwise according to the ordering of properties in the lens definition, as illustrated in Figure 8 (right). Property values revert back to their original state when the lens moves away from the resource. All these visual modifications, including color and position changes, are smoothly animated thanks to the underlying graphical toolkit's animation capabilities [27], thus keeping the user's cognitive load low following the principles of perceptual continuity.

Fresnel core formatting instructions are interpreted as customizations of the original layout and rendering of nodes and links in the diagram. For instance, nodes representing `foaf:image` property values can be rendered by fetching the actual image from the

Web, as illustrated in Figure 8 (right). The default labels of nodes and arcs can be customized using `fresnel:label` instructions. In case a resource is the subject of multiple statements involving the same property or properties defined as `fresnel:mergeProperties`, the arcs and nodes representing these statements can be merged as a single arc and node with all values within that node, optionally separated by text as specified in `fresnel:contentBefore`, `fresnel:contentAfter` and related formatting instructions.

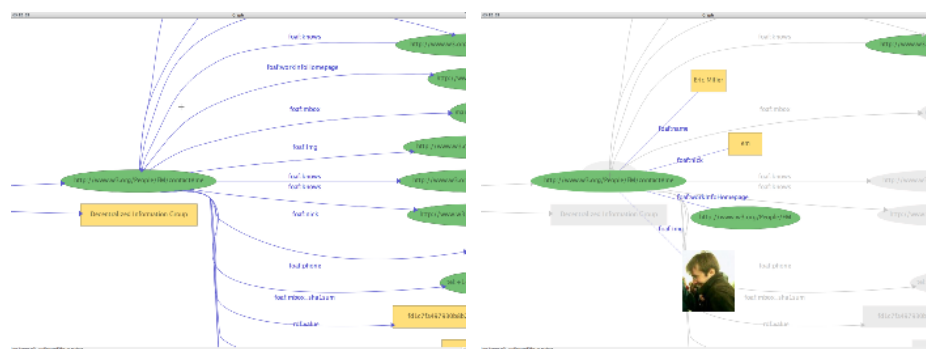


Fig. 8. Zoomed-in view of a `foaf:Person` resource in IsaViz: default presentation (left) and rendered with a Fresnel lens (right)



Fig. 9. Editing a lens (left) and visualizing the result (right) in Cardovan

Cardovan is IBM's implementation of Fresnel lenses (see Figure 9). Written in Java, Cardovan renders lenses with the SWT graphical user interface toolkit. Cardovan is similar to other implementations in that it uses a subset of CSS to specify the layout of lens components on the screen. A remarkable feature of Cardovan is that it allows users to modify a lens in place. Users can add new properties to the lens, modify property values, and rearrange the physical layout of the properties displayed, though it is not

170 E. Pietriga et al.

a full WYSIWYG Fresnel lens designer. The project is still in its early stages, but is functional and is already being used for internal projects at IBM.

5 Conclusion

We have given an overview of Fresnel, a browser-independent, extensible vocabulary for modeling Semantic Web presentation knowledge. Fresnel has been designed as a modularized, declarative language manipulating selection, formatting, and styling concepts that are applicable across representation paradigms and output formats. We have presented applications implementing Fresnel core modules while based on different representation and navigation paradigms, thus substantiating the claim that Fresnel can be used to model presentation knowledge that is reusable across browsers and other Semantic Web visualization tools.

Although core modules have been frozen for the time being, the Fresnel vocabulary remains a work in progress as new extension modules meeting special needs are being developed (e.g., for describing the *purpose* of lenses and for *editing* information). Extension modules are not necessarily aimed at being application- and paradigm-independent, as they might not be relevant in all cases; but their inclusion in Fresnel provides users with a unified framework for modeling presentation knowledge. Another field for future work is enabling Fresnel formats and lenses to be retrieved transparently from the Web so that RDF browsers could query the Web for display knowledge about previously unknown vocabularies.

The development of Fresnel is an open, community-based effort and new contributors are welcome to participate in it. More information can be found on its Web site <http://www.w3.org/2005/04/fresnel-info/>.

Acknowledgments

We would like to thank Stefano Mazzocchi, Stephen Garland, David Huynh, Karun Bakshi, Hannes Gassert, Jacco van Ossenbruggen, Dennis Quan, Lloyd Rutledge, Rob Gonzalez and Rouben Meschian for their valuable input to the design of the Fresnel vocabulary, their contributions to the discussions on the Fresnel mailing list, and work on Fresnel implementations.

References

1. SIMILE: Longwell RDF Browser (2003-2005) <http://simile.mit.edu/longwell/>.
2. Huynh, D., Mazzocchi, S., Karger, D.: Piggy Bank: Experience the Semantic Web Inside Your Web Browser. In: Proceedings of the 4th International Semantic Web Conference (ISWC). (2005) 413–430
3. Steer, D.: BrownSauce: An RDF Browser. <http://www.xml.com/pub/a/2003/02/05/brownsauce.html> (2003) XML.com.
4. Rutledge, L., van Ossenbruggen, J., Hardman, L.: Making RDF Presentable: Selection, Structure and Surfability for the Semantic Web. In: Proceedings of the 14th international conference on World Wide Web. (2005) 199–206

5. Kalyanpur, A., Parsia, B., Hendler, J.: A Tool for Working with Web Ontologies. In: Proceedings of Extreme Markup Languages. (2004)
6. Pietriga, E.: IsaViz: A Visual Authoring Tool for RDF. <http://www.w3.org/2001/11/IsaViz/> (2001-2006)
7. SIMILE: Welkin. <http://simile.mit.edu/welkin/> (2004-2005)
8. Quan, D., Huynh, D., Karger, D.R.: Haystack: A Platform for Authoring End User Semantic Web Applications. In: 2nd International Semantic Web Conference (ISWC). (2003) 738–753
9. mc schraefel, Smith, D., Owens, A., Russell, A., Harris, C.: The evolving mSpace platform: leveraging the Semantic Web on the Trail of the Memex. In: 16th ACM Conference on Hypertext and Hypermedia. (2005) 174–183
10. Berners-Lee, T., Chen, Y., Chilton, L., Connolly, D., Dhanaraj, R., Hollenbach, J., Lerer, A., Sheets, D.: Tabulator: Exploring and Analyzing linked data on the Semantic Web. In: Proceedings of the 3rd Int. Semantic Web User Interaction Workshop, Athens, USA (2006)
11. ASF: The Apache Cocoon Project. <http://cocoon.apache.org> (2005)
12. Carroll, J.J., Stickler, P.: TriX: RDF triples in XML. In: In the International Journal on Semantic Web and Information Systems, Vol.1, No.1, Jan-Mar 2005. (2005)
13. Quan, D., Karger, D.: Xenon: An RDF Stylesheet Ontology. general@simile.mit.edu mailing list attachment (2004)
14. Huynh, D.: Haystack's User Interface Framework: Tutorial and Reference. <http://haystack.lcs.mit.edu/documentation/ui.pdf> (2003)
15. Pietriga, E.: Semantic Web Data Visualization with Graph Style Sheets. In: Proceedings of the ACM Symposium on Software Visualization (SoftVis'06), Brighton, UK (2006)
16. Bizer, C., Lee, R., Pietriga, E.: Fresnel - Display Vocabulary for RDF. <http://www.w3.org/2005/04/fresnel-info/manual-20050726/> (2005)
17. FOAFers: Friend-of-a-Friend (FOAF). <http://www.foaf-project.org/> (2001)
18. Erdmann, T.I., Bizer, C.: Horus RDF Browser. <http://www.wiwiiss.fu-berlin.de/suhl/bizer/rdfapi/tutorial/horus/> (2005)
19. Berners-Lee, T.: Primer: Getting into RDF & Semantic Web using N3. <http://www.w3.org/2000/10/swap/Primer.html> (2005)
20. Prud'hommeaux, E., Seaborne, A.: SPARQL Query Language for RDF. <http://www.w3.org/TR/rdf-sparql-query/> (2005)
21. Cohen-Boulakia, S., Froidevaux, C., Pietriga, E.: Selecting Biological Data Sources and Tools with XPR, a Path Language for RDF. In: Pacific Symposium on Biocomputing (PSB), Maui, Hawaii. (2006) 116–127
22. Clark, J., DeRose, S.: XML Path Language (XPath) version 1.0. <http://www.w3.org/TR/xpath> (1999)
23. W3C: Resource Description Framework (RDF): Concepts and Abstract Syntax. <http://www.w3.org/TR/rdf-concepts/> (2004)
24. Pietriga, E.: Fresnel Selector Language for RDF. <http://www.w3.org/2005/04/fresnel-info/fsl-20050726/> (2005)
25. Furnas, G.W.: A fisheye follow-up: further reflections on focus + context. In: CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems, ACM Press (2006) 999–1008
26. Bier, E.A., Stone, M.C., Pier, K., Buxton, W., DeRose, T.D.: Toolglass and magic lenses: the see-through interface. In: SIGGRAPH '93: Proc. of the 20th conference on Computer graphics and interactive techniques, ACM Press (1993) 73–80
27. Pietriga, E.: A Toolkit for Addressing HCI Issues in Visual Language Environments. In: IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC'05), Dallas, USA (2005) 145–152

CHI 2007 Proceedings • Input Techniques

April 28-May 3, 2007 • San Jose, CA, USA

Pointing and Beyond: an Operationalization and Preliminary Evaluation of Multi-scale Searching

Emmanuel Pietriga^{1,2}
emmanuel.pietriga@inria.fr

Caroline Appert^{2,1}
appert@lri.fr

Michel Beaudouin-Lafon^{2,1}
mbl@lri.fr

¹INRIA
Bât 490 - Orsay, F-91405, France

²LRI - Univ. Paris-Sud & CNRS
Bât 490 - Orsay, F-91405, France

ABSTRACT

A number of experimental studies based on domain-specific tasks have evaluated the efficiency of navigation techniques for searching multi-scale worlds. The discrepancies among their results call for a more generic framework similar in spirit to Fitts' reciprocal pointing task, but adapted to a task that significantly differs from pure pointing. We introduce such a framework based on an abstract task and evaluate how four multi-scale navigation techniques perform in one particular multi-scale world configuration. Experimental findings indicate that, in this context, pan & zoom combined with an overview is the most efficient technique of all four, and that focus + context techniques perform better than classical pan & zoom. We relate these findings to more realistic situations, discuss their applicability, and how the framework can be used to cover a broad range of situations.

Author Keywords

Multi-scale interfaces, Searching task, Controlled Experiment, Focus + Context, Overview + Detail, Zoom

ACM Classification Keywords

H. Information Systems H.5 Information Interfaces and Presentation H.5.2 User Interfaces (H.1.2, I.3.6)

INTRODUCTION

Multi-scale interfaces (also called Zoomable User Interfaces or ZUIs) have generated a growing interest over the past decade as a powerful way of representing, navigating and manipulating large sets of data. A number of multi-scale navigation techniques have been designed and implemented, ranging from the original pan & zoom [20] to various focus+context techniques [4, 6, 27]. Up until now, the efficiency of these techniques has been evaluated with two kinds of experimental studies: usability studies based on domain-specific tasks and controlled experiments based on multi-scale versions of Fitts' pointing paradigm.

The usability studies that relied on domain-specific tasks such as searching for items on geographical maps [15], comparing hierarchical data structures [18], or reading textual documents [16] have typically produced inconclusive and sometimes contradictory results. More precisely, experimental findings varied from experiment to experiment, but since application domains varied dramatically, neither these findings can be compared nor generalized. Such results are to be expected since the performance of a given technique is indeed dependent on its context of use [1].

A better understanding of the fundamental aspects of multi-scale navigation could help explain – or even predict – such results, therefore saving valuable time and allowing better exploration of novel techniques. Fitts' pointing paradigm provides such a fundamental tool for exploring and understanding the elementary task of reaching a known target as fast as possible. Originally devised to study pointing in the real world [8], it has been used repeatedly in HCI for evaluating a variety of pointing techniques and devices [3, 2, 23]. Fitts' law has proven remarkably robust, to the point of being used as part of an ISO standard for pointing devices [25]. Fitts' pointing task has also been used with multi-scale interfaces and it has been shown that Fitts' law still applies for pointing targets with pan & zoom [10]. In particular, it has been shown that Fitts' paradigm could address navigation, not just pointing, in interfaces that require scrolling [14] or zooming [10].

While Fitts' pointing paradigm is very powerful, it models a very specific task: that of reaching a target whose location is known to the user. However, this scenario only captures one of several navigation tasks in multi-scale worlds. Users might only have partial information about the target's location and appearance, thus requiring them to search for potential targets and get more details about each one until the actual target is identified. Consider for example a user searching for Brisbane on a multi-scale world map, only knowing that it is a large Australian city. The strategy first consists in zooming towards Australia to then inspect each large city one by one, zooming in to discover that it is not the right one, zooming out, maybe as far as the whole continent, and zooming in to the next potential target until the right city is found. Exploring large spaces in search of a particular target differs from pure pointing, as it requires users to perform additional motor actions to identify the target.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CHI 2007, April 28 - May 3, 2007, San Jose, California, USA.

Copyright 2007 ACM 978-1-59593-593-9/07/0004...\$5.00.

CHI 2007 Proceedings • Input Techniques

April 28-May 3, 2007 • San Jose, CA, USA

In the same way as Fitts' reciprocal pointing task operationalizes the task of reaching a known target, we propose in this paper to operationalize the above search task in a way that is easily amenable to controlled experiments. We then evaluate how four multi-scale navigation techniques perform in one particular configuration of a multi-scale world: classical pan & zoom, overview + detail, and two focus + context techniques, namely distortion (graphical fisheye) lenses and a variation on the DragMag image magnifier which, to the best of our knowledge, has not yet been evaluated. Our results indicate that in this context overview + detail outperforms the other three, and that the two focus + context techniques outperform classical pan & zoom. However, this multi-scale world configuration is only one particular case in a range of situations. We discuss the limits of this preliminary study, describe the design space that is covered by our abstract search task and present an environment that we have developed to help explore this design space.

RELATED WORK

A number of experimental studies have compared the performance of different multi-scale navigation techniques and have reported contrasted results. Classical pan & zoom was compared with fisheye and overview + detail on high-level cognitive tasks involving electronic documents [16]: writing an essay after having explored a document, and finding answers to questions within that document. Classical pan & zoom was the least efficient technique; participants read faster using the fisheye; they wrote better essays using the overview + detail, but took more time to answer questions. In North and Shneiderman's experiment [19], participants had to browse the database of U.S. states and counties to answer questions using a detail-only scrollable interface or an overview + detail interface. The overview + detail interface outperformed the detail-only interface by 30-80% depending on the task. However, in another study [18], pan & zoom or overview + detail were not significantly different when participants had to navigate a large node-link representation and make topological comparisons. On the contrary, Hornbaeck et al. [15] have reported that their overview + detail interface was more efficient than pan & zoom. This, however, was true for only one of the two geographical maps that participants had to explore in their experiment.

The findings of these experiments show that the use of domain-dependent tasks makes it difficult to get consistent results that can be generalized, even more so when they require a significant amount of cognitive effort from the participants. Identifying and isolating lower-level, domain-independent tasks can help reach more generalizable results. From a motor perspective, one recurring task performed by users of multi-scale interfaces is to search for targets among sets of objects by navigating through space and scale. This article describes an experimental setup for the controlled evaluation of various interaction techniques considered as appropriate for the task of searching a multi-scale world.

OPERATIONALIZING MULTI-SCALE SEARCHING

Studying a task through a controlled experiment requires operationalizing it, i.e., defining it as a function of variables

of interest (independent variables) that researchers can act upon to collect measures (dependent variables). The pointing task is a well-known example in the field of HCI, initially operationalized in psychology by Fitts [8]: to study the performance of pointing techniques, researchers act on the index of difficulty (ID) variable and measure the movement time on a reciprocal pointing task. We seek to operationalize multi-scale searching in a similar way. In a multi-scale world, users navigate and look at objects until they find the target. Users have to navigate in both space and scale to a position that reveals enough details about each object, in order to decide whether it is the target or a distractor. Initially, users make a blind choice of a "potential target" at a high scale and navigate to it to acquire enough information. If it is a distractor, they have to navigate to another object, typically by zooming-out, panning, then zooming-in [10].

Since we are interested in studying the performance in time and error rate to find a target according to the required "quantity" of exploration from a purely motor perspective, we abstract the representation from any semantic or topological relationship among objects that could help participants identify the target in an uncontrolled manner (for instance, having reasonable knowledge of the geography of Russia could help locate Saint Petersburg once Moscow has been found on a map, or knowing that Chicago is on the shores of lake Michigan would reduce the area to be explored significantly). To quantify exploration, our experimental setup consists of a multi-scale world containing a set of n objects, one of them being the target and the others distractors. We define the "quantity" of exploration as the number k of distractors that users have to visit before finding the target: the larger the number of visited distractors, the larger the quantity of exploration. k is probabilistically dependent on n : the larger the number of objects, the higher the probability of having a large number of objects to visit before reaching the target. We control this parameter by forcing participants to visit a predefined number of objects before finding the target; if we chose a priori which object is the target, participants could find it immediately by chance, or on the contrary they could spend a lot of time searching for it, and this uncontrolled factor would have a significant impact on our measurements. We design our experiment to ensure that the target is the k^{th} object visited, no matter the order of exploration chosen by each participant. The system thus has to know i) when objects are seen by the participant, and ii) whether or not enough detail is displayed about these objects in order to differentiate the target from distractors. Making the system aware of these two pieces of information in a fully reliable manner requires answering the following two questions:

- What minimal scale provides enough information? This depends on visual acuity, which is user-dependent.
- In which region of the screen and for how long should an object be displayed to consider it seen? Assuming that the user visually scans the whole screen is too strong an hypothesis, and probably an unfounded one. Also, if only part of an object is in the viewport, the system cannot know for sure whether or not the user has seen it.

CHI 2007 Proceedings • Input Techniques

April 28-May 3, 2007 • San Jose, CA, USA

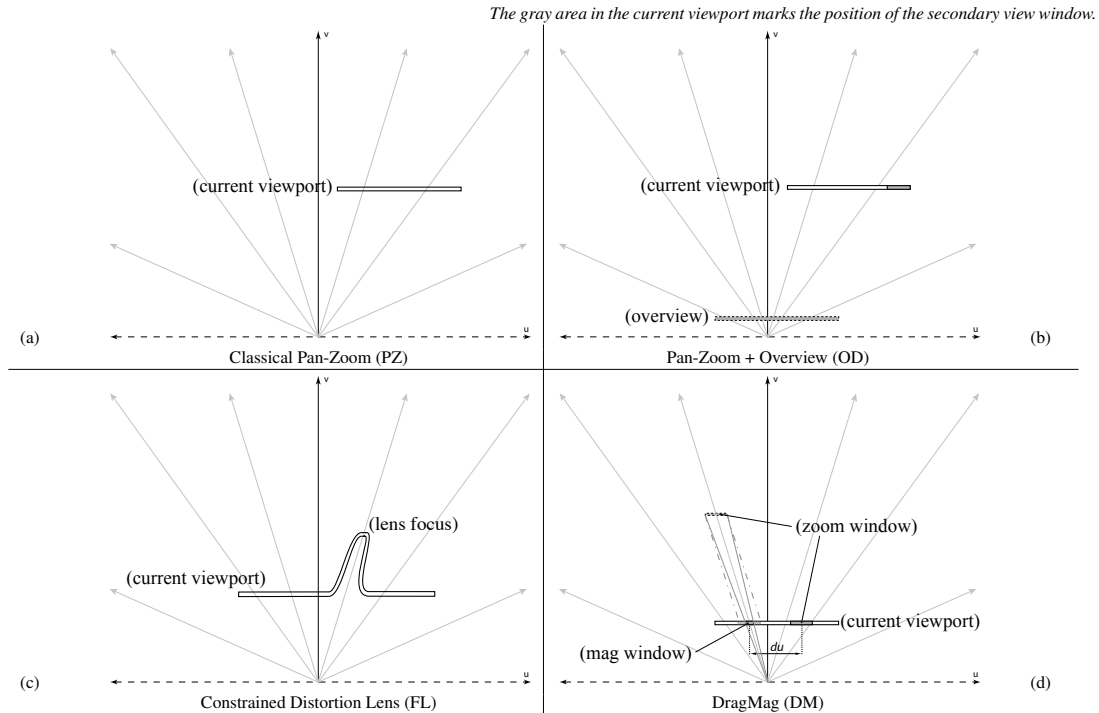


Figure 1. Representation of multi-scale interaction techniques in space-scale diagrams.

We address these problems as follows. First, we set a minimum scale (*minScale*) at which the user can collect enough information to detect a target: all objects seem identical except for the target, which reveals a different piece of information when displayed at or above *minScale*. In order to avoid differences among participants, all objects are displayed identically at all scales until the user explicitly asks to reveal the disambiguating piece of information. This explicit “unveiling” action is available only when the scale is *minScale* or more. Second, we make sure that the user cannot reveal several objects simultaneously. Once an object has been revealed, the user has to process the information and, provided that the object is the target, take an additional explicit action to tell the system that this object is the target.

While we cannot be sure that participants actually look at targets when unveiling them, it is in their own interest to do so in order to perform the search task as fast as possible. Therefore we believe that this design operationalizes a realistic search task without having to use more complex devices such as eye trackers. Before presenting a first experiment based on this task, we introduce the multi-scale navigation techniques that we have tested.

MULTI-SCALE INTERACTION TECHNIQUES

Many representation and navigation techniques have been proposed to interact with multi-scale worlds, some being variations on others. For our study, we narrowed down the possibilities to four techniques, chosen to be representative

of the most widespread and/or efficient techniques currently available. Figure 1 illustrates these techniques using space-scale diagrams [9].

The first technique is the classical *Pan & Zoom* (Figure 1-a). In order to get more detail about specific elements of the representation, users have to move the entire viewport both in space and scale, respectively by panning and zooming. No contextual information is provided; this method is therefore prone to user disorientation.

One way to address disorientation consists in using overview + detail techniques. One such technique, *Pan & Zoom + Overview* (Figure 1-b), enhances classical *Pan & Zoom* by providing users with an inset containing a representation of the region surrounding the area currently seen in the main viewport at a lower scale. The overview is located inside the main viewport, typically in one of the four corners. The goal is to minimize the visual interference caused by occlusion of the elements in focus, but this introduces the problem of divided attention [22].

In overview + detail representations, more screen real-estate is dedicated to the focus than to the context. Conversely, focus + context techniques allocate more screen real-estate to the context than to the focus. We selected two techniques that we consider relevant to the multi-scale searching task: constrained distortion lenses [6] and a variation on the original DragMag Image Magnifier [27].

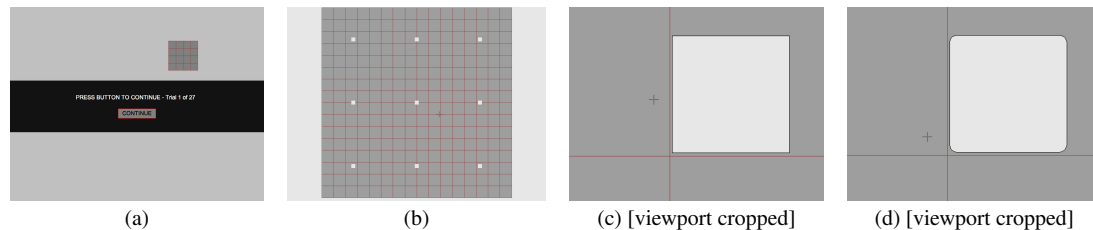


Figure 2. Storyboard: (a) start of trial, (b) navigation to the set of objects, (c) inspection of an object (before unveiling), (d) after unveiling the target

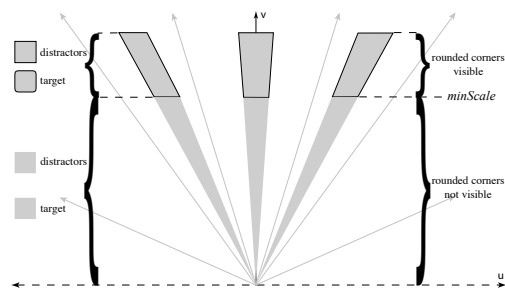


Figure 3. Space-scale diagram of the scene used in the experiment

Constrained distortion lenses (Figure 1-c) provide a detail-in-context representation through the local magnification of a region of the screen (the focus of attention). This focus region is integrated in the surrounding context by distorting the representation in the transition region. The distortion is defined by a drop-off function (see [6] for more details). We chose a Gaussian profile as it provides a smooth transition between focus and distortion, and between distortion and context. Our lens also features a flat top because many tasks require the focal region not to be distorted [5]. The *in situ* magnification of these lenses solves the problem of divided attention but introduces a distortion that can cause recognition problems.

The *DragMag* (Figure 1-d) can be considered a special case of fisheye lens often called Manhattan lens, featuring a perpendicular drop-off function. There is no distorted region between the focus and the context, but as a result the region immediately surrounding the focus is occluded. To address this issue, the focus region is translated by a user-controlled offset (du in Figure 1-d). This results in the occlusion of another region of the context, which is often considered less important than the immediate surroundings of the focus. However, this reintroduces the problem of divided attention encountered with overview + detail representations, and the occlusion can be more cumbersome to handle than with the overview.

EXPERIMENT

We conducted a 4x9 within-subject controlled experiment to compare the efficiency of these four techniques on one instantiation of the multi-scale search task introduced earlier.

Task

The task consisted in finding a target among a set of objects as quick as possible while minimizing the number of errors. The virtual scene contained nine light gray squares organized into a 3x3 grid layout and embedded inside a large, darker gray square. We used a grid layout so participants would easily know where the potential targets were. This regular layout also prevented performance to be biased by uneven traveled distances between trials of the same rank k . A dark red grid was superimposed on the display in order to minimize desert-fog [17] (see Figure 2-b). The grid was adaptive to scale, i.e., new grid lines would fade in when zooming in and some grid lines would fade out when zooming out so that the display would always contain a reasonable number of grid lines. All nine objects had square corners except for the target which had rounded corners. The rounded corners could only be seen when the target was displayed at a large enough scale, called *minScale* (Figure 3). Participants thus had to zoom in onto each square in order to find out whether it was the actual target or not. Zooming in was not sufficient however: once *minScale* was reached, a black border was displayed around the square in focus. Participants could then use the space bar to unveil the object: this would permanently reveal whether the object was the target (round corners) or not. Note that this “unveiling” step does not affect the ecological validity of our task since it penalizes all techniques equally.

Figure 2 shows a storyboard of the task: participants started each trial by pressing a button located at the center of the screen (Figure 2-a). The view was initialized so that the region containing potential targets (dark gray area) was not centered on the screen, requiring participants to reach the region by panning and zooming. The goal was both to better simulate a multi-scale navigation & search task and to avoid a learning effect with respect to the participant's initial move. Participants had to navigate to that region (Figure 2-b) and then inspect each object more closely by magnifying it using the current navigation technique (Figure 2-c). Participants were allowed to zoom-in further, but zooming in too far would have the object fill the display and make it impossible to find out if it was the target. Once *minScale* was reached for an object, participants could unveil that object by pressing the space bar. The object's border flashed green for 400 milliseconds, informing the participant that the object had actually been unveiled. If the object's corners remained square, this meant that the object was not the target and participants had to navigate to the next potential target using the current navigation technique. If, on the contrary, the object's

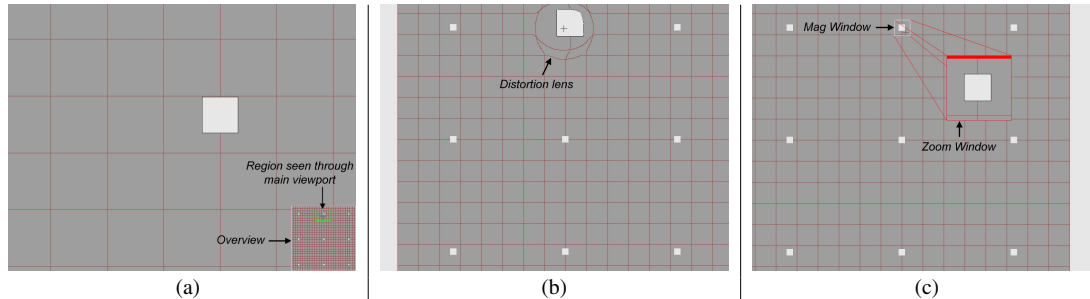


Figure 4. Unveiling an object with (a) pan-zoom + overview [OD], (b) a constrained distortion lens [FL], (c) a DragMag [DM]

corners became rounded (Figure 2-d), participants had to hit the **F1** key to tell the system that they had identified the target and end the trial. Note that figures 2-c and 2-d are cropped versions of the viewport, aimed at illustrating the actual display size of objects at *minScale* on the monitor used for the experiment.

Participants were instructed to go as fast as possible to complete a trial (i.e., between hitting the **Continue** button and hitting **F1**), but they were allowed to rest between successive trials. They were also instructed to minimize the number of visits to the same object and the number of misses, i.e., hitting **F1** when the object was not the target. Such misses terminated the trial and were counted as errors.

Techniques

The first independent variable we manipulated in our experiment was the technique. The first technique was pan & zoom (PZ). Participants could pan the view by moving the mouse while holding the left mouse button, and zoom in/out by rotating the mouse wheel. These three degrees of freedom could be controlled simultaneously. The magnification factor per wheel step was tuned so as to get an average zooming speed of 8x per second, as advocated in [15]. With this technique, participants panned & zoomed the entire view to get enough details about each object. Each of the other three techniques allowed participants to pan & zoom using the above commands.

The second technique was overview + detail (OD). The region seen through the main viewport was represented by a bright green rectangle in the inset containing the overview (see Figure 4-a). This rectangle could be dragged, resulting in changing the content of the main viewport. With these additional two degrees of freedom, participants could do fine-grain panning in the main viewport and coarse-grain panning in the overview. The representation in the overview was dynamic: it was not necessarily showing all objects in the virtual world, as it followed the camera associated with the detailed view in space and scale when the scale difference between the overview and the detailed view was larger than a factor of 24. The overview implemented by Google Maps¹ demonstrates such a behavior.

¹<http://maps.google.com>

The third technique featured a constrained distortion lens, also called graphical fisheye lens (FL). It allowed for magnification of the region around the mouse cursor (see Figure 4-b). We used a 100-pixel radial lens defined by a Gaussian drop-off function and the $L(2)$ distance metric [6] with a 60-pixel radius flat top. The lens was not activated at the start of a trial. Participants could activate it by clicking the left mouse button, and deactivate it by clicking the right mouse button. The lens was always centered on the mouse cursor. When the lens was active, participants were still able to pan the context by dragging outside the lens with the left mouse button. The default magnification factor within the flat top was set to 4 times the scale factor of the context (the scale factor in the lens focus is always defined relative to that of the context, since the context can itself be panned and zoomed). Participants could change the lens' magnification by using the mouse wheel, within the limits of twice and twelve times the scale factor of the context. This technique therefore featured five degrees of freedom (2D panning of context, 2D panning of lens focus, and either the lens magnification factor or the context scale factor depending on whether the lens is active or not). Lens activation and deactivation were both animated by smoothly increasing the lens' magnification factor from 1.0 to its default value (4.0) over a period of 300 milliseconds for the sake of perceptual continuity [24]. The lens thus seemed to "emerge" from the flat surface when activated, and flatten itself when deactivated.

The last technique was inspired by the DragMag Image Magnifier (DM), but interaction with the windows differed significantly from the original prototypes [27]. Figure 4-c shows the two windows composing the DragMag: the **mag window** outlines the region magnified in the **zoom window**. Participants could activate and deactivate the DragMag by clicking on the right mouse button. The mag window would then appear centered around the mouse cursor, the zoom window being offset by a default distance of 200 pixels to the southeast of the mag window. As with the previous technique, both DragMag activation and deactivation were smoothly animated over a period of 300 milliseconds, with the zoom window "emerging" from the mag window. Participants could drag the mag region, thus changing the content of the zoom window; they could also drag "through" the zoom window for small scale adjustments, though this feature was not very useful in the context of the experiment.

CHI 2007 Proceedings • Input Techniques

April 28-May 3, 2007 • San Jose, CA, USA

Participants could also move the zoom window by dragging the thick bar at its top. This feature was useful to reveal objects occluded by the zoom window. The mouse wheel was used to control magnification. Operating the mouse wheel while the cursor was in the zoom window controlled that window's magnification factor. Operating the mouse wheel anywhere outside this window controlled the scale of the context. The technique therefore featured six degrees of freedom (the context scale factor and the zoom window magnification factor could both be controlled when the DragMag was active). The default magnification factor in the zoom window was 4 times the scale factor of the context, as for the distortion lens. The zoom window was not resizable.

For the purpose of comparing the techniques, the overview of OD, the lens of FL, and the zoom window of DM all used the same amount of screen real-estate: a 200 x 200 pixels region, which represented 4.5% of the total available display area.

Predictions

Our predictions were as follows:

- **Time is linearly dependent on the rank k of the target.** We hypothesized that, whatever the technique, the user has to navigate to inspect objects one by one and that each navigation incurs the same cost. Since the cost of revisiting an object is fairly high, we hypothesized that the number of revisits would be very small. Therefore the overall task completion time should be linearly dependent on the "quantity" of exploration, i.e., the target's rank k in the sequence of visited objects.
- **Focus + Context (FL, DM) and Overview + Detail (OD) outperform classical Pan & Zoom (PZ).** With PZ, navigating from one object to the next typically consists in zooming out to acquire the next object then zooming in and panning to magnify it. With DM, FL and OD, it simply consists in moving the focus onto the next object. Since the position of the focus can be controlled from the context, we hypothesized that the zoom-out/pan/zoom-in sequence of PZ would take more time than relocating the focus within the context.
- **Overview + Detail (OD) outperforms Focus + Context (DM and FL).** With OD, DM and FL, navigating from object a to object b consists in moving the focus from a to b . This movement can be seen as a pointing task. With DM and FL, pointing is achieved by relocating the focus (i.e., the DragMag window or the lens' focus region) while with OD, pointing is achieved by relocating the detailed view. According to Guiard et al. [11], such pointing tasks are view pointing tasks whose Index of Difficulty depends on view size. Since the detailed view is significantly larger than the lens' focus and the DragMag's zoom window, we predicted that OD would outperform the two Focus + Context techniques (FL, DM).

Participants

Twelve unpaid adult volunteers, 11 males and 1 female, ranging from 23 to 52 years old (28 on average, with a median of

25.5), served in the experiment. Before starting, the experimenter checked that they could perceive the rounded corners at *minScale*, showing them squares with squared and rounded corners successively. The experiment was divided into four blocks, one block per technique. Before each block, participants were shown how to achieve the task using the corresponding technique. They were then asked to practice on randomly-chosen trials until they felt comfortable with the technique. The experimenter observed participants and encouraged them to keep practicing until they were familiar enough with the technique.

Apparatus

We used a Dell Precision 380 equipped with a 3 GHz Pentium D processor, an NVidia Quadro FX4500 graphics card, a 1280x1024 LCD monitor (19") and a Dell optical mouse with a scroll wheel. The program was written in Java 1.5 using the open source ZVTM toolkit [21] which features a wide range of multi-scale interaction techniques, thanks to different types of portals [20] and arbitrary distortion lenses². The application was limited to a 1080x824 window with a black padding of 100 pixels in order to accommodate instruction messages and simulate screen real-estate that would usually be taken by control and information widgets.

Counterbalancing strategy

We used a 9x4 within-subject design: we tested 9 target ranks ($k \in [1..9]$) for the 4 techniques (PZ, FL, DM and OD), i.e., $9 * 4 = 36$ conditions. Each condition was replicated 3 times so that each participant performed $9 * 4 * 3 = 108$ trials (≈ 45 minutes). The initial position of the area containing the objects was different for each of these 3 replications and was counterbalanced among blocks with a Latin square. We grouped the trials into 4 blocks, one block per technique, to minimize negative skill transfers. To minimize ordering effects, we computed four different technique orders using a Latin square and composed 4 groups of 3 participants (G_1, G_2, G_3, G_4), one group per ordering.

We also counterbalanced the presentation order of the different values of k within a block: we used a Latin square to compute 9 possible orders for presenting the values of k and concatenated 3 orders to compose a block (3 orders of 9 trials = 27 trials per block). Three block compositions (bc_1, bc_2, bc_3) were obtained through a Latin square. We mapped one block composition per participant within a group. Table 1 summarizes our counterbalancing strategy among participants. While we told participants that the target was selected randomly by the program, this was not, in fact, the case: instead, the program counted the objects being visited by the participant during the trial, and displayed the target when the k^{th} object was unveiled by the participant. This allowed us to fully control the rank variable. Note that even if the participants had known (or guessed) the actual working of the program, this would not have given them any advantage.

²The content of the lens is not a mere magnification of the original pixels, but an actual high-resolution separate rendering of the region seen through the lens, which provides more details. This mechanism also makes it possible to use semantic zooming inside the lens (see Figure 6).

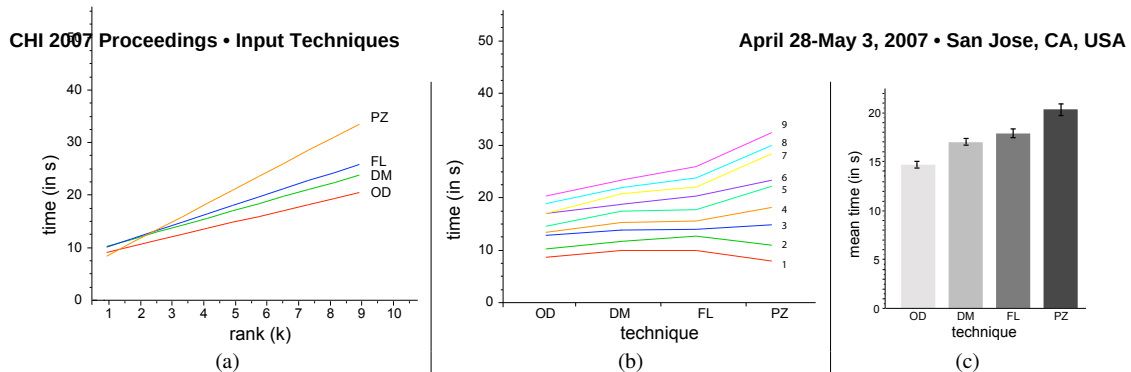


Figure 5. Fit lines for the four techniques (a), Interaction effects on time (in s) for $rank * technique$ (b), Mean completion time per technique (c)

	G_1	G_2	G_3	G_4
bc_1	S_1	S_4	S_7	S_{10}
bc_2	S_2	S_5	S_8	S_{11}
bc_3	S_3	S_6	S_9	S_{12}

Table 1. Counterbalancing strategy for the 12 participants (S_i).

	PZ	FL	DM	OD
r^2	0.84	0.84	0.81	0.80
a	3.2	2.0	1.7	1.3
b	5.1	8.2	8.5	7.8

Table 2. Correlation coefficients (r^2) and coefficients a and b ($time = a * rank + b$) for the four techniques.

Results

For each trial, the program collected the completion time, whether it was a hit or a miss, the order of visit of each object and the time at which it was unveiled. It also logged cinematic data from the cameras associated with the focus and context viewports. We also collected the participants' preferences among the techniques in a post-hoc test.

For our analyses, we first removed 14 *miss* trials (about 1%) and then 31 outliers (about 2.5% of the *hit* trials). We verified that misses and outliers were randomly distributed across participants, techniques and ranks and that there was no effect of technique presentation order on time. Learning effects were not significant for PZ ($p = 0.42$) and FL ($p = 0.75$), and were significant but moderate for DM ($p = 0.03$) and OD ($p = 0.02$).

We isolated the *rank* variable (k) by analyzing it separately for each *technique*. We computed the linear regression of time relative to the rank, treating participants as a random variable. We obtained the high correlation coefficients listed in Table 2. This supports our first prediction: completion time is linearly dependent on the rank (see Figure 5-a).

As expected, the number of revisits was extremely low (less than 1 revisit on average for each technique) and participants optimized the order in which they visited the objects so as to minimize traveled distance. Most participants explored the objects following an S-shaped pattern, some used a spiral; very few made diagonal moves, except for one participant who adopted a very erratic search pattern across all blocks (his results were nevertheless consistent with our overall findings). Table 2 also reports slopes (a) and intercepts with the y-axis (b) for each linear regression. We note that the value of b is lower for PZ. The cinematic logs explain this difference: with DM, FL and OD, participants initially spent more time adjusting the scale and position in order to optimize their future interactions. Indeed, with these techniques,

good position of the context allows participants to only pan the detailed view through the overview or the focus from the context without having to adjust the scale.

Since we have evidence that time is linearly dependent on *rank*, we now analyze *rank* as a continuous factor. Analysis of variance with the REML method for repeated measures revealed a significant simple effect on time for *rank*, i.e. k , ($F_{1,411} = 1500.5$, $p < 0.0001$), a significant simple effect on time for *technique* ($F_{3,411} = 91.6$, $p < 0.0001$) and a significant interaction effect on time for *rank * technique* ($F_{3,411} = 54.2$, $p < 0.0001$). Figure 5-b illustrates these results: the larger the rank, the larger the differences among techniques. Tukey post-hoc tests reveal that each technique is significantly different from the others: OD is the most efficient technique, followed by FL, then DM and finally PZ ($OD > DM > FL > PZ$). This supports our second and third predictions: the Overview + Detail technique outperforms the two Focus + Context techniques, which themselves outperform classical Pan & Zoom. We believe the lower performance of FL, compared with DM, could be due to the visual distortion introduced by the lens [13]. We note however that the difference between the means of these two techniques ($FL_{mean} = 18$ s., $DM_{mean} = 17$ s.) is much smaller than with the other two ($OD_{mean} = 14.8$ s., $PZ_{mean} = 21.2$ s.), as shown in Figure 5-c.

The subjective preferences we collected in the post-hoc questionnaire match these results. At the end of the experiment, participants were asked to rank the techniques according to their preference: 11 ranked PZ as the worst technique, and 9 ranked OD as the best technique.

DISCUSSION AND FUTURE WORK

The search task introduced in this article covers a range of situations where the user has to explore each potential target in a multi-scale environment. Unlike the tasks tested in us-

CHI 2007 Proceedings • Input Techniques

April 28-May 3, 2007 • San Jose, CA, USA

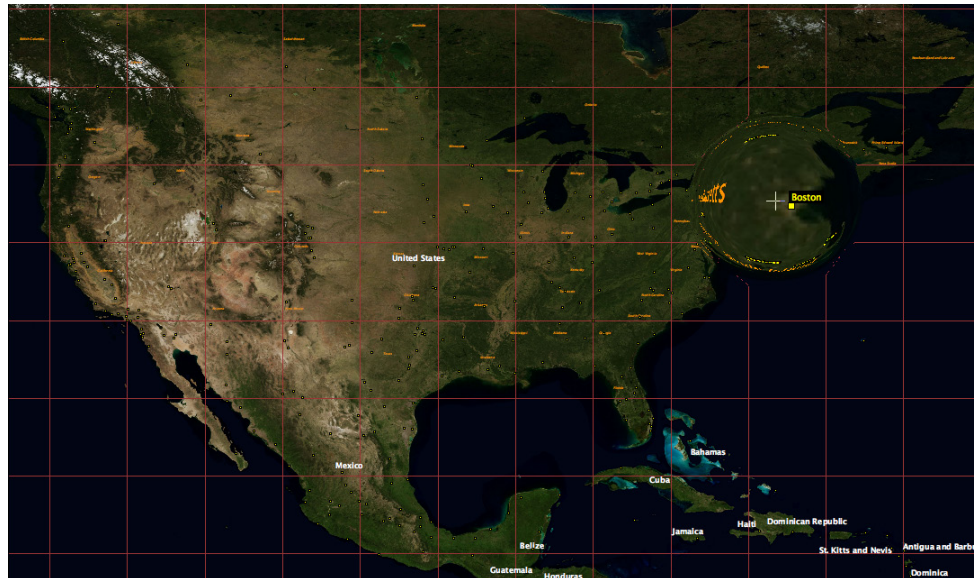


Figure 6. Looking for Boston, Massachusetts, USA with a distortion lens on a multi-scale world map

ability studies, we focus on the motor and perceptual skills and try to exclude the cognitive skills involved in searching. The goal is similar to that of Fitts' pointing paradigm and its use in HCI: to assess the limit performance of searching multi-scale worlds and to come up with predictive performance models and novel navigation techniques that improve multi-scale searching.

Our search task covers a large design space whose main dimensions are the amount of information the user has to acquire in order to decide which object is the target and the structure of the multi-scale world. Our experiment tested an extreme situation in this design space. First, the user had to look in detail at each target by navigating to it, therefore excluding the kind of visual search that occurs, e.g., in a Fitts' pointing task with distractors. Second, we used a specific configuration of the multi-scale world: a "small-world", i.e. an environment in which there exists at least one viewpoint from which all objects can be seen, that contained objects of the same relative size, i.e. same *minScale*, laid out uniformly on a grid. Therefore, the results reported in the previous section cannot be generalized to all search tasks and we need to devise a strategy to explore the design space and operationalize other situations.

Unfortunately, few theoretical models are available to help us structure this design space. While Guiard et al.'s degree of goal-directedness [12] could help quantify the amount of information that users need to recognize a target and Furnas & Bederson's space-scale diagrams [9] could help explore the structure of multi-scale worlds, neither approach is readily applicable to identify relevant points in this design space. Therefore we have developed an environment for testing re-

alistic multi-scale navigation and searching tasks in order to inform our design process.

This environment (see Figure 6) displays a multi-scale version of NASA's Blue Marble Next Generation world map [26] overlaid with geographical features such as countries, states, cities, parks and lakes. The map is 80000x40000 pixels at full resolution, or about 80x40 regular-size screens. The geographical features can be any set of localized items found in the Geo-Names³ on-line database. The environment provides a set of navigation techniques, including those tested in the study reported in this article. A variant of this environment was used to run the experiment reported in the previous section. Both versions are implemented with the ZVTM toolkit [21] and are publicly available⁴.

We conducted several pilot studies with this environment using a set of 1825 cities, 63 states and provinces, and 192 countries. Participants were asked to search for geographical features by locating first the country, then possibly the state or province and finally the city. Obviously, this task relies on cognitive skills such as the participant's geographical knowledge or contextual hints such as large water bodies. It was extremely useful however for observing users and collecting quantitative data and subjective evaluations and helped us identify interesting multi-scale world configurations.

For example, the configuration that we tested in the experiment described in the previous section corresponds to, e.g., finding a large city in Australia. Since there are only eight

³<http://www.geonames.org>

⁴<http://zvtm.sourceforge.net/eval/pb>

CHI 2007 Proceedings • Input Techniques

April 28-May 3, 2007 • San Jose, CA, USA

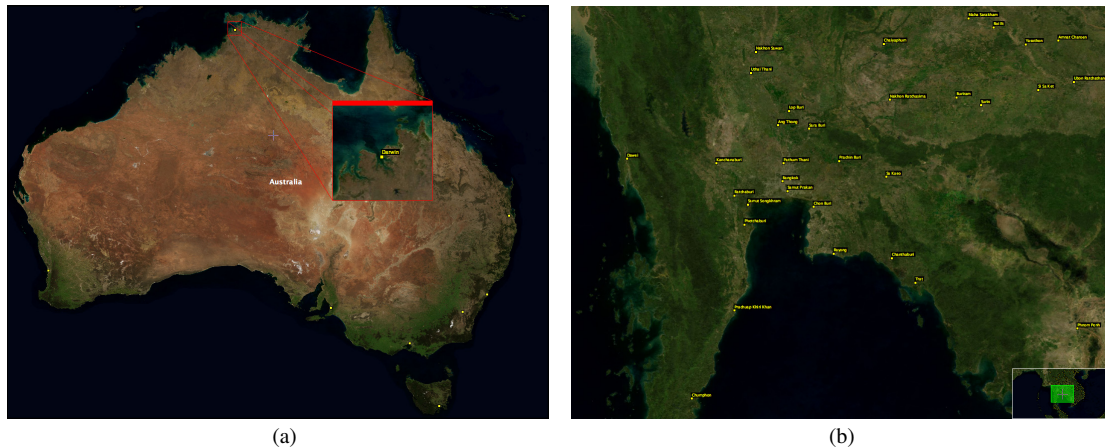


Figure 7. Exploring a sparse region with a drag-mag (a), and a dense region with an overview (b)

large cities spread over the whole continent (see Figure 7-a), the participant who does not know the geography of Australia has to zoom in onto each city. So the task consists in finding a city among a relatively small, well-identified, set of objects of the same relative size.

In this context, participants found the most useful technique to be Overview + Detail, followed by the constrained distortion lens and the DragMag. For the latter two, the commonly adopted search technique consisted in panning & zooming to make the entire continent fit the viewport (all cities could be seen from this altitude, though their names were not visible), and then activate a lens or DragMag to inspect the potential targets while keeping the context fixed. The same behavior was observed with the abstract task, as reported earlier. It is interesting to note however that the negative effects of distortion were less frequently mentioned for the geographical task than for the abstract task, probably because continuous representations such as world maps withstand distortion better than other types of representations, at least for searching.

Other observations of the participants' behavior with the geographical task have helped us identify situations that seem interesting for subsequent experiments. For instance, densely populated regions such as mainland Southeast Asia (see Figure 7-b), which feature many cities, were most commonly explored with the Overview + Detail technique because the main viewport can accommodate more cities at the scale where their names become readable (the equivalent of *minScale* defined in the abstract task), thus facilitating visual scanning.

These behavior patterns lead us to hypothesize that Overview + Detail techniques work better when exploring dense regions while Focus + Context techniques are also efficient when searching for a target among a sparse set. This may be due to the fact that visual scanning plays an important role in the former case while motor actions take precedence

over visual scanning in the latter, at least within the limits of the magnification factor of graphical fisheye lenses (usually 4 and rarely more than 8 [7]). Providing empirical evidence for this claim requires running more experiments within the framework by varying parameters such as density. Another area for future work is to test configurations in which objects have different *minScale* values, corresponding to situations where users have very limited information about the target, including the scale at which it is visible. Since such situations presumably prompt for more zooming actions than the one we tested, it is possible that the best navigation technique would be different.

SUMMARY

This paper has introduced a new framework based on an abstract searching task for multi-scale interfaces that operationalizes the situation where one has to look for the target before selecting it. We have used this framework to compare four multi-scale navigation techniques in the context of one specific multi-scale world configuration (small world, uniformly dense layout), showing that in this case a fixed overview afforded better performance than Focus + Context techniques and that traditional pan-and-zoom was the worst. These results cannot be immediately generalized to all multi-scale world configurations, and additional evaluations are required to cover a broader range of situations by varying parameters such as density, topology and the relative size of targets. Our framework allows for the systematic exploration of this design space. Moreover, the geographical environment we have developed can help identify interesting situations and formulate hypotheses about them. These situations can then easily be translated into configurations of the abstract task and tested with controlled experiments.

Acknowledgements

We wish to thank the anonymous reviewers for their insightful reviews, as well as all the volunteers who participated in our experiments.

CHI 2007 Proceedings • Input Techniques

April 28-May 3, 2007 • San Jose, CA, USA

REFERENCES

1. C. Appert, M. Beaudouin-Lafon, and W. Mackay. Context matters: Evaluating interaction techniques with the CIS model. In *Proc. of HCI 2004, Leeds, UK*, pages 279–295. Springer Verlag, Sept. 2004.
2. C. Appert and J.-D. Fekete. OrthoZoom scroller: 1D multi-scale navigation. In *CHI '06: Proc. Human Factors in Computing Systems*, pages 21–30. ACM Press, 2006.
3. R. Balakrishnan. "Beating" Fitts' law: virtual enhancements for pointing facilitation. *Int. J. Hum.-Comput. Stud.*, 61(6):857–874, 2004.
4. S. K. Card, J. D. Mackinlay, and B. Shneiderman, editors. *Readings in information visualization: using vision to think*. Morgan Kaufmann Publish. Inc., 1999.
5. M. S. T. Carpendale, D. J. Cowperthwaite, and F. D. Fracchia. Making distortions comprehensible. *IEEE Symposium on Visual Languages*, pages 36–45, 1997.
6. M. S. T. Carpendale and C. Montagnese. A framework for unifying presentation space. In *UIST '01: Proc. ACM Symposium on User Interface Software and Technology*, pages 61–70. ACM Press, 2001.
7. S. Carpendale, J. Ligh, and E. Pattison. Achieving higher magnification in context. In *UIST '04: Proc. ACM symposium on User Interface Software and Technology*, pages 71–80. ACM Press, 2004.
8. P. M. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *J. Exp. Psych.*, 47:381–391, 1954.
9. G. W. Furnas and B. B. Bederson. Space-scale diagrams: understanding multiscale interfaces. In *CHI '95: Proc. Human Factors in Computing Systems*, pages 234–241. ACM Press/Addison-Wesley, 1995.
10. Y. Guiard and M. Beaudouin-Lafon. Target acquisition in multiscale electronic worlds. *Int. J. Hum.-Comput. Stud.*, 61(6):875–905, Dec. 2004.
11. Y. Guiard, M. Beaudouin-Lafon, J. Bastin, D. Pasveer, and S. Zhai. View size and pointing difficulty in multi-scale navigation. In *AVI '04: Proc. working conference on Advanced Visual Interfaces*, pages 117–124. ACM Press, 2004.
12. Y. Guiard, Y. Du, and O. Chapuis. Quantifying degree of goal directedness in document navigation: Application to the evaluation of the perspective-drag technique. In *CHI '07: Proc. Human Factors in Computing Systems*. ACM Press, 2007. This volume.
13. C. Gutwin. Improving focus targeting in interactive fisheye views. In *CHI '02: Proc. Human Factors in Computing Systems*, pages 267–274. ACM Press, 2002.
14. K. Hinckley, E. Cutrell, S. Bathiche, and T. Muss. Quantitative analysis of scrolling techniques. In *CHI '02: Proc. Human Factors in Computing Systems*, pages 65–72. ACM Press, 2002.
15. K. Hornbæk, B. B. Bederson, and C. Plaisant. Navigation patterns and usability of zoomable user interfaces with and without an overview. *ACM Trans. Comput.-Hum. Interact.*, 9(4):362–389, 2002.
16. K. Hornbæk and E. Frøkjær. Reading of electronic documents: the usability of linear, fisheye, and overview+detail interfaces. In *CHI '01: Proc. Human Factors in Computing Systems*, pages 293–300. ACM Press, 2001.
17. S. Jul and G. W. Furnas. Critical zones in desert fog: Aids to multiscale navigation. In *UIST'98: Proc. ACM Symposium on User Interface Software and Technology*, pages 97–106, 1998.
18. D. Nekrasovski, A. Bodnar, J. McGrenere, F. Guimbretière, and T. Munzner. An evaluation of pan & zoom and rubber sheet navigation with and without an overview. In *CHI '06: Proc. Human Factors in Computing Systems*, pages 11–20. ACM Press, 2006.
19. C. North and B. Shneiderman. Snap-together visualization: a user interface for coordinating visualizations via relational schemata. In *AVI '00: Proc. working conference on Advanced Visual Interfaces*, pages 128–135. ACM Press, 2000.
20. K. Perlin and D. Fox. Pad: an alternative approach to the computer interface. In *SIGGRAPH '93: Proc. Computer Graphics and Interactive Techniques*, pages 57–64. ACM Press, 1993.
21. E. Pietriga. A Toolkit for Addressing HCI Issues in Visual Language Environments. In *IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC'05)*, pages 145–152, Dallas, USA, 2005.
22. M. Posner and S. Petersen. The attention system of the human brain. *Annual Review of Neuroscience*, 13:25–42, 1990.
23. G. Ramos and R. Balakrishnan. Zliding: fluid zooming and sliding for high precision parameter manipulation. In *UIST '05: Proc. ACM symposium on User Interface Software and Technology*, pages 143–152. ACM Press, 2005.
24. G. G. Robertson, S. K. Card, and J. D. Mackinlay. Information visualization using 3d interactive animation. *Comm. ACM*, 36(4):56–71, 1993.
25. R. W. Soukoreff and I. S. MacKenzie. Towards a standard for pointing device evaluation, perspectives on 27 years of fitts' law research in hci. *Int. J. Hum.-Comput. Stud.*, 61(6):751–789, 2004.
26. R. Stockli, E. Vermote, N. Saleous, R. Simmon, and D. Herring. The Blue Marble Next Generation - A true color earth dataset including seasonal dynamics from MODIS. Published by the NASA Earth Observatory, 2005.
27. C. Ware and M. Lewis. The DragMag image magnifier. In *CHI '95 conference companion, Human Factors in Computing Systems*, pages 407–408. ACM Press, 1995.

CHI 2009 ~ Pointing and Cursor Techniques

April 8th, 2009 ~ Boston, MA, USA

DynaSpot: Speed-Dependent Area Cursor

Olivier Chapuis^{1,3}
chapuis@lri.fr

Jean-Baptiste Labrune²
labrune@media.mit.edu

Emmanuel Pietriga^{3,1}
emmanuel.pietriga@inria.fr

¹LRI - Univ. Paris-Sud & CNRS
Orsay, France

²MIT Media Lab
Cambridge, MA, USA

³INRIA
Orsay, France

ABSTRACT

We present DynaSpot, a new technique for acquiring targets based on the area cursor. DynaSpot couples the cursor's activation area with its speed, behaving like a point cursor at low speed or when motionless. This technique minimizes visual distraction and allows pointing anywhere in empty space without requiring an explicit mode switch, thus enabling users to perform common interactions such as region selections seamlessly. The results of our controlled experiments show that the performance of DynaSpot can be modeled by Fitts' law, and that DynaSpot significantly outperforms the point cursor and achieves, in most conditions, the same level of performance as one of the most promising techniques to date, the Bubble cursor.

ACM Classification Keywords

H.5.2 Information Systems: Information Interfaces and Presentation – User Interfaces, Input Devices and Strategies

Author Keywords

Area cursor, Bubble cursor, DynaSpot, Fitts' Law

INTRODUCTION

The increase in both resolution and size of computer displays requires users of desktop interfaces based on the ubiquitous WIMP paradigm to make highly precise pointing movements to acquire small interface components over possibly long distances when using a conventional point cursor. Several techniques have been proposed to make this fundamental task easier. Many have been shown to perform better than the point cursor in experimental settings that were consisting of isolated targets on fairly sparse desktops [1, 3, 5, 10]. However, these techniques are very sensitive to the layout and density of interface components, and difficulties arise when selecting one target among multiple objects that are spatially close together. As noted by Baudisch et al. [4], non-uniform target distributions with clusters of small targets are commonplace in GUIs. In such configurations, these techniques do not provide a significant advantage and some can actually degrade performance.

Other promising techniques have been proposed recently that work better in a wider range of configurations, including many variations on expanding targets [6, 7, 20, 21], the Ninja cursor [15] and Starburst [4]. One of the most promising technique, the Bubble cursor [9], is a variation on the Area cursor [14, 26] that dynamically adapts its activation area to encompass the closest object only. This is achieved by expanding the boundaries of each target based on a Voronoi tessellation that fills the empty space surrounding each potential target thus maximizing their effective size. While this optimizes pointing performance, problems arise when considering interaction beyond the acquisition of a single interface component. First, as with several of the above-mentioned techniques, selecting a position in the “empty” space between targets requires a mode switch. Yet empty space selection is crucial to many common interactions, e.g., to select groups of objects. The mode switch solution results in “a slightly less than seamless interaction style” [2] for these essential object manipulation features [15]. Second, rapid and large changes of the bubble size in non-uniform target distributions may distract the user and hinder user acceptance [9, 17, 12], a crucial factor [2] that is sometimes overlooked.

In this paper, we present DynaSpot, a new type of area cursor that couples the cursor's activation area with its speed, as illustrated in Figure 1. The activation area grows as a function of speed up to a maximum size, typically set to a few dozen pixels, thus minimizing visual distraction. At low speed and when motionless, DynaSpot behaves as a regular point cursor, making all conventional point cursor interactions, including empty space selection, possible without the need for an explicit mode switch.

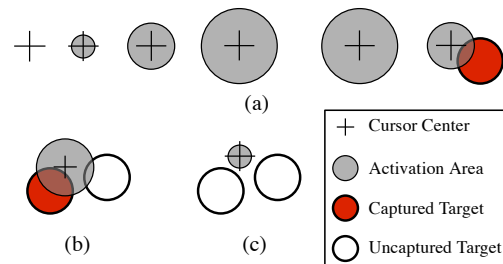


Figure 1. (a) DynaSpot's activation area is coupled to cursor speed. (b) Multiple objects intersect the area: the target closest to the cursor center is highlighted and selected. (c) Empty space selection is possible whenever the activation area is not intersecting any object.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CHI 2009, April 4 - 9, 2009, Boston, Massachusetts, USA.

Copyright 2009 ACM 978-1-60558-246-7/09/04...\$5.00.

CHI 2009 ~ Pointing and Cursor Techniques

April 8th, 2009 ~ Boston, MA, USA

After a review of related work, we discuss the design and implementation of DynaSpot, and report the results of two controlled experiments. Results show that DynaSpot significantly outperforms the point cursor and achieves levels of pointing performance similar to the Bubble cursor in most layout configurations, including densely populated scenes. We then show that its performance can be modeled with Fitts' law. We conclude with a discussion of our findings and directions for future work.

RELATED WORK

Fitts' law is the fundamental tool used to study pointing in human-computer interfaces [18, 25]. It makes it possible to predict movement time MT with the following equation:

$$MT = a + b \times \log_2\left(\frac{A}{W} + 1\right)$$

where A is the distance to the target (amplitude of movement), W the width of the target, and a, b are two coefficients determined empirically, depending on factors such as input device and population of users. Techniques developed to facilitate pointing in virtual worlds try to decrease movement time either by reducing A , increasing W , or a combination of both. We direct interested readers to a survey by Balakrishnan [2] and an overview by Grossman et al. [9] that follow this categorization to review existing techniques. In the following, we consider existing techniques from a slightly different perspective, considering not only performance but compatibility with conventional cursor interactions beyond single target acquisition, and user acceptance.

Sticky icons [26] and Semantic pointing [5] dynamically adapt the control-display ratio, slowing down the cursor as it approaches a potential target. These techniques support conventional point cursor interactions. They are, however, very sensitive to the layout and density of potential targets; while they work well in sparsely populated workspaces, intervening distractors on the path to the actual intended target in denser workspaces slow down cursor movements, possibly degrading performance compared to a regular point cursor. Cockburn and Firth [7] propose to enable the control-display adaptation on one axis only depending on the widget's orientation, thus partially solving the problem for some types of widgets such as scrollbars.

Drag-and-pop [3] reduces amplitude of movement (A) when dragging an object by temporarily bringing potential targets, closer to the cursor. As such, the technique efficiently solves one particular type of pointing-based interaction, but is not a general desktop pointing technique. Object pointing [10] takes a radical approach, ignoring the empty space between targets by making the cursor jump from one object to the nearest one in the direction of movement, thus considerably reducing A . The Delphian desktop [1] follows the same principle, taking into account peak velocity to determine the goal target, allowing to jump over potential distractors. Both techniques are very sensitive to the layout and density of objects, which can have a strong impact on the accuracy of the goal target prediction method. Lank et al., describe an enhanced endpoint prediction method [16] achieving 42% accuracy and an additional 39% of predictions falling on

an adjacent target, with one third of gesture time remaining. Still, wrong predictions can be frustrating, and the behavior of the cursor, jumping from object to object, can be annoying. By skipping empty space, these techniques do not allow the user to perform some useful point cursor interactions, such as region selection, without an explicit mode switch. Kobayashi and Igarashi propose another promising way to reduce the amplitude of movement (A) by having multiple cursor instances all synchronized with the same input device: by distributing the cursors over the screen, Ninja cursor [15] reduces the average distance to any given target using interactive, seamless disambiguation methods to activate the appropriate cursor. General point cursor interactions that require clicking in empty space are however not possible without mode switching, except for a restricted form of lasso selection.

Several techniques focus on increasing target width (W). Techniques based on lenses coupled with the cursor magnify objects but usually operate in the original, unmagnified, motor space, thus providing no actual advantage in terms of pointing facilitation [11, 23]. Ramos et al.'s Pointing lenses [24] are an exception, increasing target size in both visual and motor space for the acquisition of small targets with a stylus. Another solution consists in expanding targets dynamically when a point cursor approaches them. McGuffin and Balakrishnan [20] have found that users can still benefit from expansions that occur as late as after 90% of the movement has been completed. They were further studied in [21], and experiments by Cockburn and Brock suggest that visual expansion plays a more important role than motor expansion [6]. They also note that "*enlarged motor-spaces actually make the targets appear smaller than they really are*", as empty space around objects is actually empty in visual space only, not in motor space, meaning that it cannot be used for interactions such as region selection.

Fitts' law can accurately model pointing to thin targets using area cursors with a simple modification to the equation: instead of representing the target width, the term W represents the cursor width [14]. This implies that cursors with larger activation areas make pointing easier, but such larger areas are more likely to encompass several objects, thus creating ambiguities. These can be resolved by using a secondary point cursor [26] or by interactively adjusting the cursor area on multi-point touchpads [22]. The Bubble cursor [9] improves upon the area cursor by partitioning empty space so as to maximize the activation area of each target. Starburst [4] relies on a different partitioning of space, better adapted to non-uniform target distributions. As mentioned earlier, this optimizes pointing performance, but prevents point cursor interactions that require clicking in empty space. The Bubble cursor's growing/shrinking area has also been reported to cause visual distraction in some situations [9, 17, 12]. Several variations on the technique have been designed [17, 12], but have had limited success both in terms of performance and user acceptance. The lazy bubble [17] makes it possible to point in some areas of empty space, but these are severely limited and difficult to identify, making interactions such as region selection impractical.

CHI 2009 ~ Pointing and Cursor Techniques

April 8th, 2009 ~ Boston, MA, USA

DYNASPOT

In his survey of pointing facilitation techniques [2], Balakrishnan identifies the final acceptability of a technique by end-users as a critical measure, seen as a complement to quantitative performance measures such as selection times and error rates. The visual distraction caused by some techniques and the mode switches required by earlier-mentioned techniques hinder their acceptance for many types of applications and environments. DynaSpot has been designed to facilitate pointing while taking this more qualitative measure into account. It was not designed to perform better than all other techniques under all conditions, but to strike a balance between performance, end-user acceptance and implementation in a realistic context.

DynaSpot builds upon area cursors. It uses the dynamic characteristics of the pointer to adapt the size of the cursor's activation area and facilitate difficult pointing tasks while behaving as a conventional point cursor when appropriate, without the need for an explicit mode switch. DynaSpot takes inspiration from other techniques that have successfully made use of the cursor's dynamic characteristics, such as Speed-Dependent Automatic Zooming [13], Sigma Lenses [23] and the Speed-coupled flattening lens [11].

As shown in Figure 2, the size of the activation area (which we term *spot* from now on) starts to increase as a function of cursor speed past a given threshold, and up to a maximal size *SPOTWIDTH*. When the cursor comes to a full stop, reduction of the spot starts after a certain duration *LAG*, and takes *REDUCETIME* to complete. As with regular area cursors, the spot is made translucent so as to avoid obscuring screen information relevant to the task [26].

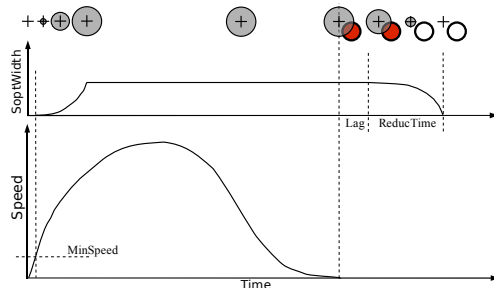


Figure 2. DynaSpot: spot width as a function of cursor speed.

A target can be selected as soon as the spot overlaps it. While early area cursor designs [14, 26] used a square shape, DynaSpot's activation area takes the shape of a circle, as does Bubble cursor's, so as to ensure that the nearest target is captured first. Still, as opposed to the latter technique, there can be situations where the spot overlaps more than one potential target, creating ambiguities regarding the one to select. To resolve such ambiguities, DynaSpot always selects the target closest to the cursor center (see Figure 1-b). This implies that the system should provide feedback about which target is currently selected (if any), as is the case for Bubble cursor. If the spot does not intersect any potential target, then the background (or "empty space") is selected, no matter the

current spot width, allowing the user to perform any action initiated by a button press in empty space, such as a region selection (see Figure 1-c).

According to Fitts' law, DynaSpot should facilitate pointing because the *potential effective width* of a target can be larger than its actual width. For instance, if we consider an isolated circular target of width W and a spot width of SW at the time of actual target selection (i.e., when clicking), then the potential effective width is $EW = W + SW$. Thus, when the user clicks on the target before the spot starts shrinking (before the end of *LAG* in Figure 2), the effective width of the target is $EW = W + \text{SPOTWIDTH}$, as illustrated in Figure 3-a. If we consider a target surrounded by other targets with empty space between them of width IS , as in Figure 3-b, then the potential effective width depends on the spot width and what we term the *interspace* between targets, IS . If $IS \leq \text{SPOTWIDTH}$, then the potential effective width is $EW = W + IS$. The optimal *SPOTWIDTH* will depend on a number of factors: interface type, display resolution, input device, but also on each user. In a typical desktop environment, a *SPOTWIDTH* between 16 and 32 pixels represents a good compromise: it is large enough to facilitate the acquisition of small targets, yet small enough to prevent size variations from causing too much visual distraction.

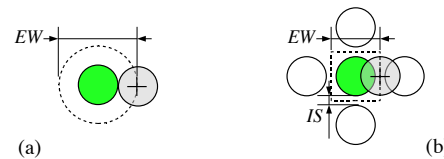


Figure 3. (a) Isolated target: potential effective width $EW = W + \text{SPOTWIDTH}$. (b) Small interspace between targets $IS \leq \text{SPOTWIDTH}$: potential effective width is $EW = W + IS$.

Speed-dependent Behavior

Figure 2 gives a general idea of the speed coupling between cursor speed and spot width. The details of this coupling play a fundamental role in the overall usability of the technique, and are described in this section. The behavior rules are as follows:

- when the cursor is moved fast enough (beyond a threshold speed $S \geq \text{MINSPEED}$ pixel.s⁻¹), the width of the spot is increased, provided that it has not yet reached its maximal value *SPOTWIDTH*;
- when the cursor comes to a full stop and does not move for a period of time equal to *LAG*, the spot shrinks to a point (1 pixel) over a period of *REDUCETIME*, provided the user does not move it again, in which case it would grow again;
- for slow movements below the speed threshold, the spot width remains constant.

Threshold speed *MINSPEED* allows the user to perform small, precise pointing movements using a conventional point cursor, without being distracted by a growing spot. When the cursor is moved faster, beyond this threshold, the spot grows, facilitating distant target acquisition. We have found 100 pixel.s⁻¹ to be a reasonable value for *MINSPEED*.

CHI 2009 ~ Pointing and Cursor Techniques

April 8th, 2009 ~ Boston, MA, USA

The transitions from point cursor to area cursor and conversely can be achieved in various ways. We tested several possibilities through trial and error, and made the following observations. The spot should grow quickly once *MIN SPEED* has been reached, but the growth profile does not seem to play an important role. We found that an exponential growth (up to *SPOT WIDTH*) by a factor of 1.2 at each input event works well.

The reduction transition, controlled by *LAG* and *REDUCTIME* (see Figure 2), is more complex because it has a direct impact on the potential effective width at the time of target selection. Higher values for both parameters should make the task easier. However, too high values imply that the user will potentially have to wait longer before she can perform interactions initiated by an implicit selection in empty space. In addition, the reduction profile applied during *REDUCTIME* also plays a role.

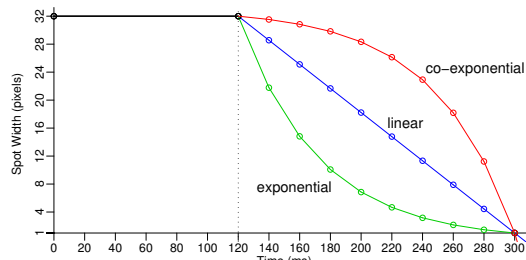


Figure 4. Spot width as a function of time for the three reduction methods (*LAG* = 120 ms, *REDUCTIME* = 180 ms).

We informally tested three methods to perform this reduction, as illustrated in Figure 4: (i) an “exponential” one where the spot width is reduced by a given percentage at each step; (ii) a “linear” one where the spot width is reduced by a given constant at each step, and (iii) a “co-exponential” one that mirrors the first method. We found that the exponential reduction yields more target acquisition errors, probably because of the abrupt transition after the *LAG* period, due to the fast reduction of the spot. The co-exponential method starts reducing the spot at a lower pace, providing a smoother transition than the linear and exponential methods.

PRELIMINARY STUDY: LAG AND REDUCTION TIME

Before comparing DynaSpot with other pointing techniques, we ran a preliminary experiment in which we formally evaluated different values of *LAG* and *REDUCTIME* for the co-exponential transition in order to fine-tune the technique.

Apparatus

We used a 3.2 GHz Pentium4 PC running X Window under Linux, equipped with an NVidia Quadro FX 1500 graphics card, a 1600 x 1200 LCD monitor (21”), and a standard optical mouse (400 dpi) with the default X Window acceleration function. Our program was written in Java using the OpenGL pipeline for graphics rendering. We carefully checked the refresh rate (50 fps), ensuring that timers were matching the lag and reduction set for each condition.

Participants

Eight unpaid adult volunteers (7 male, 1 female), from 22 to 41 year-old (average 26.6, median 24), all right-handed, experienced mouse users, served in the experiment.

Procedure and Design

The task was a simple reciprocal pointing task. The two targets were represented as circles 8 pixels in diameter, painted with a green fill color and outlined in black. They were centered horizontally, with a distance of 512 pixels between them, and were each surrounded by four distractors of the same size, painted with a white fill color and outlined in black. These four distractors were laid out so that the inter-space *IS* between a distractor and the target would always match the *SPOT WIDTH* set for the current trial, as illustrated in Figure 8-b. We focused on small targets in this preliminary study as DynaSpot is expected to be most useful in this type of configuration. The object captured by the cursor (distractor or actual target, if any) was filled with a red color. Each target had to actually be selected before proceeding to the next: clicks outside the current target were counted as errors but did not end the task.

Our experiment was a $2 \times 3 \times 3$ within-participant design. Each participant had to perform several trials using two spot widths: *SPOT WIDTH* $\in \{16, 32\}$ with three durations for both lag and reduction time: *LAG* $\in \{60ms, 100ms, 140ms\}$ and *REDUCTIME* $\in \{100ms, 180ms, 260ms\}$.

We grouped trials into two blocks, one for each spot width. Four participants started with the small DynaSpot (16 pixels) while the four others started with the larger one (32 pixels). Within a block, trials were grouped by *LAG* $\times$ *REDUCTIME* condition presented in a pseudo-random order, each sub-block containing three series of 16 reciprocal pointing tasks. The first series was used for training, allowing participants to adapt to the new parameters before we measured their performance. They were then instructed to be as accurate and as fast as possible. The 16 pointing tasks of a series had to be performed in a row, but participants were allowed to rest between trials. The first targeting task of each trial was ignored. A total of 4,320 actual pointing tasks were thus taken into account in the analysis (240 measures for each *SPOT WIDTH* $\times$ *LAG* $\times$ *REDUCTIME* condition). The experiment started with a 3 minute training session where the experimenter explained DynaSpot’s behavior and how to operate it to the participant. The experiment lasted approximately 20 minutes.

Results

Repeated measures analysis of variance reveals a significant simple effect on movement time for *SPOT WIDTH* ($F_{1,7} = 97.0, p < 0.0001$), *LAG* ($F_{2,14} = 11.5, p = 0.0011$) and *REDUCTIME* ($F_{2,14} = 7.5, p = 0.006$). The only significant interaction is for *LAG* $\times$ *REDUCTIME* ($F_{4,28} = 3.3, p = 0.0238$).

Mean movement time is 884 ms for *SPOT WIDTH* = 16 and 760 ms for *SPOT WIDTH* = 32. The *LAG* $\times$ *REDUCTIME* effect can be observed on Figure 5 (left): *LAG* seems to have an

CHI 2009 ~ Pointing and Cursor Techniques

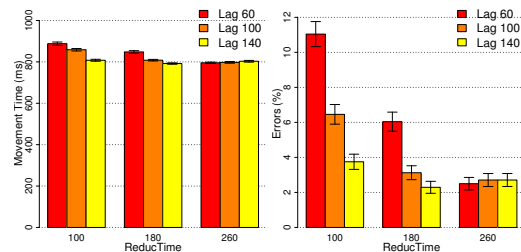


Figure 5. Movement time (left) and error rate (right) as a function of LAG, grouped by REDUCTIME.

effect for REDUCTIME equal to 100 and 180 ms but not for REDUCTIME = 260 ms. This is confirmed by post-hoc tests, which show a significant difference in mean between all LAG values for REDUCTIME = 100, between LAG 60 and 100 for REDUCTIME = 180, but no significant difference for REDUCTIME = 260. The overall error rate is 4.5%. Repeated measures analysis of variance reveals no effect on error rate for SPOTWIDTH ($F_{1,7} = 0.4$, $p = 0.522$), but a significant effect for LAG ($F_{2,14} = 10.8$, $p = 0.0014$) and REDUCTIME ($F_{2,14} = 17.7$, $p < 0.0001$). As for movement time, we observe a significant interaction for LAG $\times$ REDUCTIME only ($F_{4,28} = 3.4$, $p = 0.0227$), as illustrated in Figure 5 (right).

These results show that for a long-enough REDUCTIME, LAG can be set to any value within the considered range. For shorter REDUCTIMES, the duration of LAG has a significant effect on both movement time and error rate, and has to be chosen carefully. Overall, the fastest and least error prone condition evaluated was LAG = 140 ms and REDUCTIME = 180 ms. For our implementation of DynaSpot, we did not want the full reduction phase to last longer than 300 ms, as longer delays can be frustrating. We thus used the following values: LAG = 120 ms and REDUCTIME = 180 ms.

MAIN EXPERIMENT: DYNASpot VS. BUBBLE VS. POINT
Having fine-tuned DynaSpot's parameters, we ran a second experiment to evaluate the quantitative performance of DynaSpot and get the subjective impressions of participants. We compared two DynaSpots with different spot widths (16 and 32 pixels) against a regular point cursor, serving as a baseline, and the Bubble cursor [9], one of the most efficient general pointing techniques to date (Figure 6).

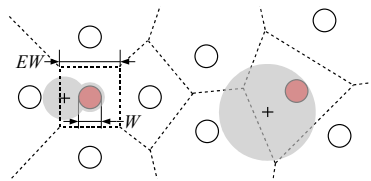


Figure 6. The Bubble cursor captures the target closest to its center. The shape of the targets are expanded to a maximal shape obtained by a Voronoi tessellation. The bubble's effective width, EW , is thus defined by the corresponding shape.

Apparatus

We used a workstation running X Window under Linux, equipped with two double core 64-bits 2.4 GHz processors, an

April 8th, 2009 ~ Boston, MA, USA

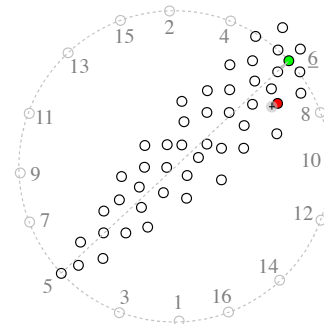


Figure 7. Sixth pointing task of a series (ISO 9241-9 circular layout)

NVidia Quadro FX4500 graphics card, a 1600 x 1200 LCD monitor (21") and a standard optical mouse (400 dpi) with the default X Window acceleration function. Our program was written in Java using the OpenGL pipeline for graphics rendering, thus ensuring a minimum frame rate of 50 fps even for large alpha-blended Bubble cursor areas (something impossible with the default Java2D rendering pipeline).

Participants

Twelve unpaid adult volunteers (all male), from 21 to 33 year-old (average 25.2, median 25), all right-handed, experienced mouse users, served in the experiment.

Task and Procedure

We followed the same general procedure as the one used by Grossman and Balakrishnan to compare Bubble cursor with object pointing and a point cursor [9]: participants had to select a target rendered as a solid green circle outlined in black, surrounded by a set of distractors. Additional distractors were placed on the path from the trial start point to the target. As illustrated in Figure 7, all distractors were the same size as the target and were rendered as black outlined circles. As in our preliminary experiment, the object captured by the cursor (if any) was painted red. The bubble cursor area and the DynaSpot disc were both rendered with a semi-transparent gray. As in the Bubble cursor paper's experiment, four main distractor targets were positioned to control the interspace IS around, and thus the effective width EW of, the goal target¹. Two were placed along the direction of movement, one on each side of the target, while the other two were placed perpendicular to the direction of movement (see Figure 3-b). The remaining distractors were laid out so as to match the density condition DD on the path to the target. For $DD = 0$, there were no additional distractors on the path to the target. For $DD = 1$, additional distractors were packed from the start point to the closest main distractor, and offset in the direction perpendicular to the line of movement by a pseudo-random length, keeping them within a 20 degree slice centered in this line of movement. Additional distractors outside this slice were placed pseudo-randomly to match the density within the slice. For $DD = 0.5$, there were half as many distractors.

¹The original Bubble cursor experiment controlled the interspace in terms of EW/W ratio (Figure 6), which we will also use here for cross-experiments comparisons.

CHI 2009 ~ Pointing and Cursor Techniques

April 8th, 2009 ~ Boston, MA, USA

We made the following adjustments to the original design. Instead of making the next target appear in an unpredictable location, we laid out all 16 targets of a trial series in a circular manner. The order of appearance followed the recommendations of the ISO 9241-9 standard forcing participants to perform pointing tasks in every direction [8]. We chose this more predictable behavior of targets, encountered in several pointing experiments, e.g., [6, 11, 23, 25], as it better simulates situations where users have a rough idea about the direction of the target they are aiming at before starting the pointing task. Each target had to actually be selected before proceeding to the next: clicks outside the current target were counted as errors but did not end the task.

Design

Our experiment was a $4 \times 3 \times 3 \times 3 \times 3$ within-participant design with the following factors: (i) four techniques $TECH$: Bubble, DynaSpot16 ($SPOTWIDTH = 16$), DynaSpot32 ($SPOTWIDTH = 32$) and Point Cursor; (ii) three target widths W : 8, 16 and 32 pixels; (iii) three amplitudes A : 256, 512 and 768 pixels; (iv) three EW/W ratios: 1.5, 2 and 3; (v) three distractor densities DD : 0, 0.5 and 1.

We grouped trials into four blocks, one per technique. Each $TECH$ block was divided into 3 sub-blocks, one per EW/W condition. Each of these sub-blocks was composed of $3 W \times 3 A$ series of 16 pointing tasks where each DD was used 5 times (the first task of a series was not recorded). An additional sub-block at the beginning (W and A random) was used for training. To counterbalance the presentation order of conditions, we computed a Latin square for $TECH$ and a Latin square for EW/W and crossed them, obtaining 12 orders, one for each participant. The order of the $W \times A$ conditions, as well as the density DD , were chosen randomly but the same order was used for each $TECH$ across participants for the 15 recorded tasks of a series.

The experiment started with a training session consisting of 4 $TECH \times 3 EW/W$ series, each with $W = 16$ and $A = 512$. The experimenter introduced each technique to the participant during the first series of each corresponding $TECH$ block; the two remaining blocks being used as actual training. For the series actually recorded, participants were instructed to be as accurate and as fast as possible. The 16 pointing tasks of a series had to be performed in a row, but participants were allowed to rest between series. A total of 19,440 actual pointing tasks were thus taken into account in the analysis (60 measures for each unique condition). The experiment lasted approximately 45 minutes.

Combined Width and Hypotheses

One of the main factors used in the experiment comparing Bubble cursor to other techniques [9] was the EW/W ratio (effective width of the goal target by its actual width). The combination of this and factor W controls the distance between the goal target and the four distractors surrounding it. This abstraction, well-adapted to the former experiment, is however not best suited to analyze the different conditions with DynaSpot.

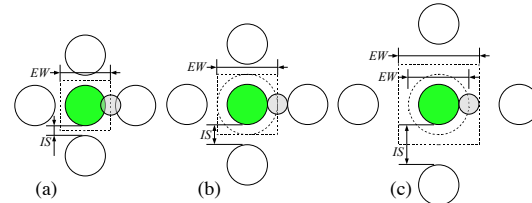


Figure 8. (a) $IS < \text{DynaSpot Width}$: Bubble and DynaSpot have the same EW . (b) $IS = \text{DynaSpot Width}$: Bubble and DynaSpot have the same EW , but different effective target shapes. (c) $IS > \text{DynaSpot Width}$: Bubble's EW is greater than DynaSpot's EW .

Since the distractors are uniformly placed around the target, the ratio can be expressed in terms of interspace IS between the goal target and the distractors: $EW = W + IS$ and $(EW/W) = (W + IS)/W$. As illustrated in Figure 8, this formulation helps identify the three main conditions for DynaSpot: the spot width can be (a) larger than, (b) equal to, or (c) smaller than, the interspace IS . The factors W and EW/W can be grouped into one factor that we call the *combined width* CW . In the remainder of this paper, we use the following notation for each pair of conditions $W \times EW/W$:

$$(BW, \underline{DW}_{16}, DW_{32}, W)$$

where BW is the Bubble cursor's effective width, $\underline{DW}_{16}$ and DW_{32} are the potential effective widths for the two DynaSpot sizes, and W is the target's width (which corresponds to the effective width of the target for the point cursor). The factors described in the previous section yield nine combined widths CW , listed in Table 2. When the interspace IS is equal to one of the DynaSpot $TECH$'s potential effective width (case (b) of Figure 8), we underline the corresponding DynaSpot (16 and 32 pixels).

Our main hypothesis is that for a given combined width CW , the effective width EW for each technique should determine the performance ordering among techniques: if the effective width for technique a is larger than for technique b (for a given combined width), then a should be faster than b (for this combined width). When the effective widths of two techniques are equal, we do not expect to find significant differences in terms of performance. However, we expect a performance degradation when DynaSpot is at its limit effective width (underlined width in the CW notation). Indeed, in this particular case, the spot reduction and small intersection between the target and the spot may forbid the user to use the full potential effective width of DynaSpot. Additionally, we hypothesize that density DD will have a similar effect on point cursor and both DynaSpots, as the behavior of all three techniques is not directly impacted by density. On the contrary, we expect a performance degradation for low densities in the case of Bubble cursor, consistent with Grossman and Balakrishnan's observations regarding visual distraction in this condition [9].

Results

Results of the repeated measures analysis of variance are reported in Table 1. We verified that there was no effect of $TECH$ presentation order and observed that learning effects

CHI 2009 ~ Pointing and Cursor Techniques

April 8th, 2009 ~ Boston, MA, USA

Factors	DF	DFDen	F	p
TECH	3	33	115.5	< 0.0001
CW	8	88	509.0	< 0.0001
DD	2	22	28.5	< 0.0001
A	2	22	409.6	< 0.0001
TECH × CW	24	264	12.6	< 0.0001
TECH × DD	6	66	9.0	< 0.0001
TECH × A	6	66	2.3	0.0470
CW × DD	16	176	5.6	< 0.0001
CW × A	16	176	2.1	0.0085
DD × A	4	44	1.8	0.1435
TECH × CW × DD	48	528	1.0	0.3746
TECH × CW × A	48	528	1.2	0.1628
TECH × DD × A	12	132	2.3	0.0123
CW × DD × A	32	352	1.1	0.2637
TECH × CW × DD × A	96	1056	1.2	0.0904

Table 1. Results of the ANOVA for MT ~ TECH × CW × DD × A.

were not significant. As expected CW and A have a significant effect on movement time MT. We also observe an effect of TECH on movement time. Mean movement time is 976 ms for Point cursor, 831 ms for Bubble, 819 ms for DynaSpot16 and 791 ms for DynaSpot32. However, the ANOVA also reveals significant interactions: TECH × CW, TECH × DD and TECH × A. A thorough comparison between techniques must thus take into account combined width, density of distractors, and amplitude.

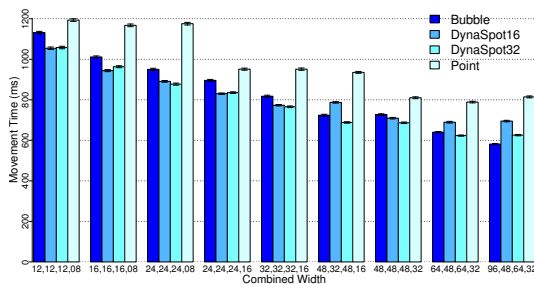


Figure 9. Mean movement time per TECH, grouped by CW.

Tukey HSD test ($\alpha = 0.05$)			
CW	All DD conditions	DD = 0	DD = 1
(12,12,12,08)	P < D32,D16 ; B < D32,D16	-	B < D32,D16
(16,16,16,08)	P < B,D32,D16	B < D32	-
(24,24,24,08)	P < B,D16,D32	B < D32	-
(24,24,24,16)	P < B,D32,D16	B < D32,D16	-
(32,32,32,16)	P < B,D16,D32	B < D16,D32	-
(48,48,48,16)	P < D16,B,D32 ; D16 < D32	-	D16 < B
(48,48,48,32)	P < B,D16,D32	-	-
(64,48,64,32)	P < D16,B,D32	-	-
(96,48,64,32)	P < D16,D32,B ; D16 < B	-	D32 < B

P = Point cursor, B = Bubble, DX = DynaSpotX

Table 2. Significant differences for mean movement time MT between TECH, by CW. The two rightmost columns show how the results are modified if we restrict our analysis to distractor densities 0 and 1.

Figure 9 shows the mean movement time for each TECH by combined width CW. Table 2 gives the results of the Tukey HSD *post-hoc* test for differences in mean between techniques by combined width (where $a < b$ means that TECH b is significantly faster than TECH a). The test shows that Bubble and DynaSpot are both significantly faster than Point and that there is little difference between Bubble and DynaSpot.

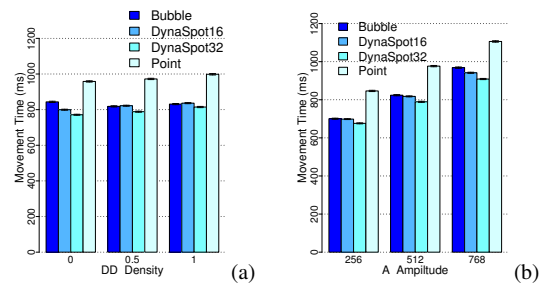


Figure 10. (a) Movement time for each TECH grouped by distractor density. (b) Movement time for each TECH grouped by amplitude.

Figure 10-a shows mean movement time for each technique grouped by distractor density. We see that movement time increases as density increases for Point cursor and DynaSpot, but not for Bubble cursor. For Bubble cursor, a post-hoc Tukey test reveals that it is faster for DD=0.5 than for DD=0, confirming the results of [9]. The test also reveals that each of DynaSpot16, DynaSpot32 and Point cursor is faster for DD=0 than for DD=1. Moreover, as shown in Table 2, Bubble cursor is slower than DynaSpot for DD=0 in most conditions where the effective widths are equal, while Bubble cursor is faster than DynaSpot for DD=1 when the Bubble's effective width is larger than the DynaSpot's effective width.

Figure 10-b shows mean movement time for each technique grouped by movement amplitude. We see that the difference between Bubble cursor and DynaSpot increases with amplitude. A post-hoc Tukey test shows that DynaSpot32 is faster than Bubble for an amplitude of 768, but no such difference is detected for smaller amplitudes. Moreover, removing the data for which DD=0 makes this significant difference disappear (a cause of the TECH × DD × A interaction).

These results show that the effective width determines the performance ordering among techniques only under certain conditions regarding distractor density. Our hypothesis is thus only partially verified. Distractor density affects Bubble cursor performance, especially for large movement amplitudes. As predicted, a significant degradation is observed when DD=0, i.e., when the bubble's envelope varies most during movement, causing visual distraction. Finally, distractor density also affects Point cursor and DynaSpot in a similar way, degrading performance as it increases.

Regarding errors, we find an overall error rate of 6.5%. Repeated measures analysis of variance shows a significant effect on error rate for TECH ($F_{3,33} = 11.6$, $p < 0.0001$), CW ($F_{8,88} = 8.3$, $p < 0.0001$) and A ($F_{2,22} = 5.7$, $p = 0.0097$). Interestingly, there is no significant effect of DD ($F_{2,22} = 1.4$, $p = 0.262$). Error rate was 9.7% for Point cursor, 6.5% for Bubble cursor, and 4.9% for both DynaSpot16 and DynaSpot32. As usual in pointing task experiments, error rate decreases as the (effective) width grows and the amplitude decreases. Again, we find a significant interaction between TECH and CW ($F_{24,264} = 2.6$, $p < 0.0001$). Figure 11 shows error rate for each TECH grouped by combined width CW.

CHI 2009 ~ Pointing and Cursor Techniques

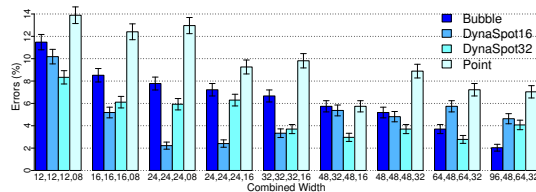


Figure 11. Error rate for each TECH grouped by combined width CW.

Removing errors trials or taking the time of the first click (instead of first *successful* click) does not change the results. We checked the number of outliers by counting the number of trials where the time is 3 standard deviations away from the mean movement time (by participant, technique, combined width, and amplitude). The data contains only 0.99% such outliers; 87% of them are errors, and none of them is more than 3.6 standard deviations away. Again the analysis without these outliers yields the same results.

Performance results for Point and Bubble cursors are consistent with those in [9]. However, our participants perform faster overall: 10.6% faster for Bubble cursor and 9.6% faster for Point cursor. This can be explained by the use of a ratio of 1.5 instead of 1.3 for the smallest value of EW/W , a larger error rate in our experiment, and by the details of the task: the location of the next target in our case was predictable, whereas it was not in [9].

Qualitative Results

Participants were asked to rank the techniques by subjective preference in a post-hoc questionnaire. All participants ranked DynaSpot (either 16 or 32) as their preferred technique, followed by the other DynaSpot in second. Only two participants chose another technique than the other DynaSpot as the second best. One ranked the Bubble cursor second, the other DynaSpot third and Point last, while the other participant ranked the Point cursor second, the other DynaSpot third, and Bubble cursor last. Most participants complained about the visual distraction caused by the Bubble cursor envelope's strong variations under certain conditions, leading seven of them to rank that technique last. This is again consistent with earlier results [9, 12]. For instance, one participant said that "Bubble cursor is distracting when the target is far away because the bubble has a big size".

Fitts' Law and Effective Width(s)

Figure 12 plots movement time as a function of IDE , the index of difficulty computed with the target's potential effective width. We take the mean for each combined width, amplitude and technique, fitting 27 points for each technique. Table 3 gives the intercept, the slope and the adjusted r^2 for both IDE and ID , the latter being computed using the actual width of the target. We see that using the effective width yields higher r^2 values and improves the fit. When fitting all the data, we obtain the equation $MT = 85 + 181 \cdot IDE$ with an adjusted r^2 of 0.962 (for 108 points). This shows that the potential effective width for DynaSpot provides a "definition" of the width appropriate for applying Fitts' Law.

April 8th, 2009 ~ Boston, MA, USA

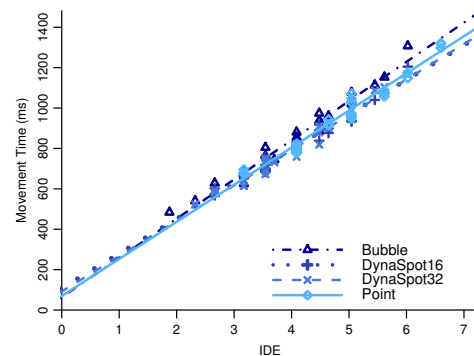


Figure 12. Linear fit: index of difficulty computed with effective width.

Models Techniques \	Using width W			Using effective width EW		
	$MT = a + b \cdot ID$			$MT = a + b \cdot IDE$		
	a	b	Adj. r^2	a	b	Adj. r^2
Bubble	-96	188	0.855	65	194	0.970
DynaSpot16	104	145	0.894	108	171	0.966
DynaSpot32	2	160	0.857	88	176	0.973
Point	69	183	0.969	69	183	0.969

Table 3. Linear fit: intercept, slope and adjusted r^2 using ID or IDE .

Figure 13 shows the position of user clicks relative to the target, and the potential/effective target width for one combined width: (64,48,64,32). We observe that clicks are scattered across a larger area for Bubble cursor than for DynaSpot32, even though both have the same effective width for this combined width. We explain this by the fact that, for DynaSpot32, this corresponds to the case described in Figure 8-b, with the interspace equal to the spot's width, preventing users from fully taking advantage of the effective width. Interestingly, we observe that users do not use the full potential of the Bubble cursor either, as there are very few clicks in the corners of the target's activation area.

In an effort to formalize these observations, we measured the distance to the center of the target for all clicks by combined width, and analyzed the 95% quantile of these distances. As expected, effective widths are reflected in these distances. But other observations can be made. For instance when DynaSpot is at its limit potential effective width (underlined conditions in CW), as for DynaSpot32 in Figure 13, we do find a significant difference between DynaSpot and Bubble cursor, as observed above, but also between DynaSpot32 and DynaSpot16 when the latter is at its limit potential effective width. Another interesting observation is that none of the 95%-quantile distances are larger than the effective width, confirming our initial observation that the corner of the Bubble's activation area are seldom used.

Another type of "effective width", that we call the *a posteriori* effective width and denote W_e , was introduced by Crossmann in his 1956 doctoral dissertation and advocated by MacKenzie and others in the field of HCI [18, 25, 27]. This *a posteriori* effective width comes from the idea of performing an "adjustment for accuracy": the width of the target is corrected so that, under certain hypotheses, the data

CHI 2009 ~ Pointing and Cursor Techniques

April 8th, 2009 ~ Boston, MA, USA

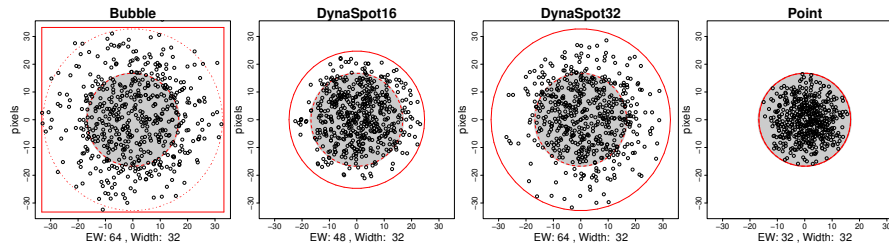


Figure 13. Position of user clicks (after a coordinates change to match a right-to-left horizontal target acquisition) for each technique for combined width CW (64,48,64,32) – Bubble effective width: 64, DynaSpot16 (resp., DynaSpot32) max. effective width: 48 (resp., 64), and target width: 32.

gives raise to an error rate of 4%. A priori, this normalization process leads to more robust results, allowing for better comparisons between experiments. In the following, we check that this definition of effective width can be used to model DynaSpot movement time.

We refer the reader to [25] for details about the computation of W_e . This involves removing outliers, using the time at first button press, computing by participant and full condition, with $W_e = 4.133 \cdot sd$ where sd is the standard deviation of the oriented distance from the click to the target's center divided by $\sqrt{2}$. Mean values of W_e for each technique $TECH$ at each combined width CW are given in the table below.

CW	Bubble	DynaSpot16	DynaSpot32	Point
(12,12,12,08)	15.37 (+0.36)	13.89 (+0.21)	13.94 (+0.22)	11.51 (+0.52)
(16,16,16,08)	20.04 (+0.32)	17.12 (+0.10)	17.26 (+0.11)	11.47 (+0.52)
(24,24,24,08)	27.31 (+0.19)	20.38 (-0.24)	26.41 (+0.14)	11.30 (+0.50)
(24,24,24,16)	26.27 (+0.13)	23.53 (-0.03)	25.38 (+0.08)	18.54 (+0.21)
(32,32,32,16)	36.15 (+0.18)	29.85 (-0.10)	32.68 (+0.03)	19.25 (+0.27)
(48,32,48,16)	52.41 (+0.13)	30.74 (-0.06)	43.35 (-0.15)	17.22 (+0.11)
(48,48,48,32)	50.03 (+0.06)	46.31 (-0.05)	48.12 (+0.00)	36.20 (+0.18)
(64,48,64,32)	68.61 (+0.10)	48.51 (+0.02)	57.19 (-0.16)	34.82 (+0.12)
(96,48,64,32)	91.57 (-0.07)	45.80 (-0.07)	59.96 (-0.09)	35.69 (+0.16)

Zhai's index of occupation [27], $I_u = \log_2(W_e/EW)$, is given in parentheses. I_u indicates the degree to which the participants over-utilize (positive I_u) or under-utilize (negative I_u) the potential effective target. We observe that the index of occupation is systematically higher for Point cursor, and then for Bubble cursor, and that it globally decreases as the width grows. This can be explained by the error rate (see Figure 11) especially for Point cursor. When comparing Bubble cursor and DynaSpot, which have similar error rates, this confirms that participants better use the full effective target width with Bubble rather than with DynaSpot.

The counterpart of W_e , the *a posteriori* effective amplitude A_e , is computed as the mean of the distance from the movement start point to the point where the user clicks. We can compute the effective index of difficulty:

$$ID_e = \log_2(A_e/W_e + 1)$$

Table 4 gives Fitts' law equation parameters for ID_e , and the throughput in $bit \cdot s^{-1}$, computed using either the slope of Fitts' equation, or the formula recommended in [25]. This throughput has the advantage of taking the intercept into account and to be less dependent both on the ID range used [25] and on the users' nominal pointing speed [19]. Thus, it

	$MT = a + b \cdot ID_e$			TP = 1000/b	TP mean of mean [25]
	a	b	adj. r^2		
Bubble	46	186	0.825	5.38 (5.15)	5.09
DynaSpot16	85	161	0.781	6.21 (5.85)	5.56
DynaSpot32	51	169	0.809	5.92 (5.68)	5.55
Point	49	179	0.756	5.59 (5.46)	5.34
All	57	174	0.792	5.75 (5.52)	5.38

Table 4. Fitts' law equation parameters for ID_e and throughputs.

may be a good index of performance to compare techniques among papers. Note that the r^2 values obtained here do not look as good as with the other method, but in this case we fit 324 points by technique, which are subject to participant performance variability.

DISCUSSION AND FUTURE DIRECTIONS

The results of our experiments are very encouraging. They show that DynaSpot provides an average speed-up of 18% over a conventional point cursor, and that for equivalent effective widths it achieves the same level of performance as the Bubble cursor, one of the more promising techniques to date. DynaSpot is slightly more efficient for low object densities and slightly less efficient for high ones. But most importantly, DynaSpot provides these quantitative performance benefits without departing too much from the conventional point cursor technique. This has at least three significant practical consequences. First, end-users are more likely to adopt the new technique in their daily use of GUIs because DynaSpot behaves "almost like" a point cursor and does not cause a strong visual distraction. Second, DynaSpot is compatible with all point cursor interactions such as region selections initiated by clicking in empty space *without requiring an explicit mode switch*. Finally, implementing DynaSpot does not require significant changes to existing GUI frameworks to support the technique: we implemented support for DynaSpot in the ZVTM Java toolkit² in less than 500 lines of code, and a lazy version of the technique³ was implemented in the Metisse windowing system⁴, relying solely on the accessibility API to make the technique work across unmodified applications.

Quantitative and theoretical analyses of performance results show that DynaSpot performance can be modeled with both

²<http://zvtm.sf.net>

³Which only needs to know the position and shape of interface components at click time, but does not feature target highlighting.

⁴<http://insitu.lri.fr/metisse>

CHI 2009 ~ Pointing and Cursor Techniques

April 8th, 2009 ~ Boston, MA, USA

the *a priori* and *a posteriori* effective widths. Somewhat unexpectedly, DynaSpot proves to be on a par with Bubble cursor in most targeting situations. If Bubble cursor's effective width is sufficiently larger than DynaSpot's, then Bubble cursor is faster. However, this happens mostly in configurations where the bubble cursor size is likely to vary dramatically, causing visual distractions that both hinder performance and user acceptance of the technique.

As future work we would like to evaluate area selection. We can predict what should happen with DynaSpot: (i) if empty space between targets is sufficiently large compared to the maximum spot size, the time it takes to initiate a selection in empty space should be similar to the time it takes with a point cursor; (ii) in a dense layout, we expect DynaSpot to be penalized because of the lag-reduction time (300ms). In this particular situation, an explicit mode-switching mechanism might represent an interesting compromise. DynaSpot would then have to be compared, for dense layouts, to Bubble and Area cursors augmented with such an explicit mode-switch, but also to an augmented DynaSpot featuring both time-based (implicit) and explicit mode-switching. We also plan to investigate the use of speed coupling in other pointing techniques, as this seems to be an efficient way of adapting a technique's behavior.

REFERENCES

1. T. Asano, E. Sharlin, Y. Kitamura, K. Takashima, and F. Kishino. Predictive interaction using the delphian desktop. In *Proc. UIST '05*, 133–141. ACM, 2005.
2. R. Balakrishnan. "Beating" Fitts' law: virtual enhancements for pointing facilitation. *Inter. J. of Hum.-Comput. Stu.*, 61(6):857–874, 2004.
3. P. Baudisch, E. Cutrell, D. Robbins, M. Czerwinski, P. Tandler, B. Bederson, and A. Zierlinger. Drag-and-pop and drag-and-pick: Techniques for accessing remote screen content on touch and pen-operated systems. In *Proc. Interact '03*, 57–64, 2003.
4. P. Baudisch, A. Zotov, E. Cutrell, and K. Hinckley. Starburst: a target expansion algorithm for non-uniform target distributions. In *Proc. AVI '08*, 129–137. ACM, 2008.
5. R. Blanch, Y. Guiard, and M. Beaudouin-Lafon. Semantic pointing: improving target acquisition with control-display ratio adaptation. In *Proc. CHI '04*, 519–526. ACM, 2004.
6. A. Cockburn and P. Brock. Human on-line response to visual and motor target expansion. In *Proc. GI '06*, 81–87, 2006.
7. A. Cockburn and A. Firth. Improving the acquisition of small targets. In *Proc. British HCI '03*, 181–196, 2003.
8. S. A. Douglas, A. E. Kirkpatrick, and I. S. MacKenzie. Testing pointing device performance and user assessment with the ISO 9241, part 9 standard. In *Proc. CHI '99*, 215–222. ACM, 1999.
9. T. Grossman and R. Balakrishnan. The bubble cursor: enhancing target acquisition by dynamic resizing of the cursor's activation area. In *Proc. CHI '05*, 281–290. ACM, 2005.
10. Y. Guiard, R. Blanch, and M. Beaudouin-Lafon. Object pointing: a complement to bitmap pointing in GUIs. In *Proc. GI '04*, 9–16, 2004.
11. C. Gutwin. Improving focus targeting in interactive fish-eye views. In *Proc. CHI '02*, 267–274. ACM, 2002.
12. M. Hertzum and K. Hornbæk. Input techniques that dynamically change their cursor activation area: A comparison of bubble and cell cursors. *Int. J. Hum.-Comput. Stud.*, 65(10):833–851, 2007.
13. T. Igarashi and K. Hinckley. Speed-dependent automatic zooming for browsing large documents. In *Proc. UIST '00*, 139–148. ACM, 2000.
14. P. Kabbash and W. Buxton. The "prince" technique: Fitts' law and selection using area cursors. In *Proc. CHI '95*, 273–279. ACM/Addison-Wesley, 1995.
15. M. Kobayashi and T. Igarashi. Ninja cursors: using multiple cursors to assist target acquisition on large screens. In *Proc. CHI '07*, 949–958. ACM, 2008.
16. E. Lank, Y.-C. N. Cheng, and J. Ruiz. Endpoint prediction using motion kinematics. In *Proc. CHI '07*, 637–646. ACM, 2007.
17. J. Laukkanen, P. Isokoski, and K.-J. Räihä. The cone and the lazy bubble: two efficient alternatives between the point cursor and the bubble cursor. In *Proc. CHI '08*, 309–312. ACM, 2008.
18. I. S. MacKenzie. Fitts' law as a research and design tool in human-computer interaction. *Hum.-Comput. Interact.*, 7:91–139, 1992.
19. I. S. MacKenzie and P. Isokoski. Fitts' throughput and the speed-accuracy tradeoff. In *Proc. CHI '08*, 1633–1636. ACM, 2008.
20. M. McGuffin and R. Balakrishnan. Acquisition of expanding targets. In *Proc. CHI '02*, 57–64. ACM, 2002.
21. M. J. McGuffin and R. Balakrishnan. Fitts' law and expanding targets: Experimental studies and designs for user interfaces. *ACM Trans. Comput.-Hum. Interact.*, 12(4):388–422, 2005.
22. T. Moscovich and J. F. Hughes. Multi-finger cursor techniques. In *Proc. GI '06*, 1–7, 2006.
23. E. Pietriga and C. Appert. Sigma lenses: focus-context transitions combining space, time and translucence. In *Proc. CHI '08*, 1343–1352. ACM, 2008.
24. G. Ramos, A. Cockburn, R. Balakrishnan, and M. Beaudouin-Lafon. Pointing lenses: facilitating stylus input through visual-and motor-space magnification. In *Proc. CHI '07*, 757–766. ACM, 2007.
25. R. W. Soukoreff and I. S. MacKenzie. Towards a standard for pointing device evaluation: Perspectives on 27 years of Fitts' law research in HCI. *Int. J. Hum.-Comput. Stud.*, 61(6):751–789, 2004.
26. A. Worden, N. Walker, K. Bharat, and S. Hudson. Making computers easier for older adults to use: area cursors and sticky icons. In *Proc. CHI '97*, 266–271. ACM, 1997.
27. S. Zhai, J. Kong, and X. Ren. Speed-accuracy trade-off in Fitts' law tasks: on the equivalency of actual and nominal pointing precision. *Int. J. Hum.-Comput. Stud.*, 61(6):823–856, 2004.

CHI 2009 ~ Understanding Graphs

April 9th, 2009 ~ Boston, MA, USA

Topology-Aware Navigation in Large Networks

Tomer Moscovich¹ **Fanny Chevalier¹** **Nathalie Henry²** **Emmanuel Pietriga^{2,3}** **Jean-Daniel Fekete²**
 tomer@lri.fr chevalie@lri.fr nhenry@lri.fr pietriga@lri.fr fekete@lri.fr

¹Microsoft Research-INRIA Joint Centre
Orsay, France

²INRIA
Orsay, France

³LRI - Univ. Paris-Sud & CNRS
Orsay, France

ABSTRACT

Applications supporting navigation in large networks are used every day by millions of people. They include road map navigators, flight route visualization systems, and network visualization systems using node-link diagrams. These applications currently provide generic interaction methods for navigation: pan-and-zoom and sometimes bird's eye views.

This article explores the idea of exploiting the connection information provided by the network to help navigate these large spaces. We visually augment two traditional navigation methods, and develop two special-purpose techniques. The first new technique, called "Link Sliding", provides guided panning when continuously dragging along a visible link. The second technique, called "Bring & Go", brings adjacent nodes nearby when pointing to a node. We compare the performance of these techniques in both an adjacency exploration task and a node revisiting task. This comparison illustrates the various advantages of content-aware network navigation techniques. A significant speed advantage is found for the Bring & Go technique over other methods.

ACM Classification Keywords

H. Information Systems H.5 Information Interfaces and Presentation H.5.2 User Interfaces (H.1.2, I.3.6)

Author Keywords

Interaction techniques, content-aware, graph visualization, document navigation

INTRODUCTION

Applications supporting navigation in large networks are used every day by millions of people. They include road map navigators such as Google maps [10], flight route visualization systems such as Delta Air Line Route Map [6] and network visualization systems using node-link diagrams [1, 13]. These applications currently provide generic interaction techniques for navigation: pan-and-zoom and sometimes bird's eye views.

However, for large networks, some important tasks related to the network's topology are not efficiently supported by existing techniques. For example, using Google maps, exploring a long route often involves panning over long portions of a highway with no exits. Zooming out or using a bird's eye view is possible, but some highway exits are difficult to distinguish from roads passing over or under the highway, so an exit can be missed. The same problem arises in network visualization systems where nodes are connected by links that can be long and cross many other links. Following a specific link can take a long time without zooming out, but zooming out makes it difficult to trace a link when other links cross it at a shallow angle.

The problem of panning and zooming, or using a bird's eye view, becomes even more difficult when using a small screen to view a network. On a PDA or smart phone, the input device may not have dedicated zooming controls, and only a small footprint is available for panning gestures, making panning over a long route slow and tedious.

In this article we explore several techniques to improve navigation in such network-related scenarios by using topological information in addition to geometric information. Two techniques are simply visual enhancements of common spatial navigation methods, while two are novel approaches that test different trade-offs between topological and spatial navigation. The first, Link Sliding, allows users to slide along a link to its destination, while the other, Bring & Go, brings all possible destinations within the users view, and automatically transports the user to the selected point.

We begin by introducing the Link Sliding and Bring & Go techniques. We then present a controlled experiment that compares them with visually augmented pan-and-zoom and bird's eye view navigation for three fundamental navigation tasks. We finally discuss implications to the design of systems for large-network visualization.

RELATED WORK

In their survey on Graph Visualization and Navigation, Herman et al. [14] cite four methods for navigating large networks: pan-and-zoom, space distortion techniques such as fisheye views, topological methods such as Furnas's "Generalized Fisheye Views" [8] and layout techniques to dynamically change the layout of the network according to the user's navigation.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CHI 2009, April 4-9, 2009, Boston, Massachusetts, USA.

Copyright 2009 ACM 978-1-60558-246-7/09/04...\$5.00.

CHI 2009 ~ Understanding Graphs

April 9th, 2009 ~ Boston, MA, USA

Scrolling, Panning and Zooming

Scrolling consists of using a widget, such as a scroll-bar, to control the viewport. Panning uses direct manipulation of the viewport, usually coupled with zooming. A lot of work has been dedicated to improving navigation using scrolling, panning, and zooming, particularly in facilitating navigation in very large spaces, or the navigation to off-screen targets.

Navigation in Large Spaces

When the space is large compared to the viewport size—say more than ten times the size—navigation can take a substantial amount of time, particularly for exploratory tasks. Early on, scroll-bars were proposed as a means for traveling through large documents. However, they suffer from several limitations. They show only the size of the viewport relative to the size of the document, and give no information regarding off-screen content. When the size of the document is large compared to the viewport—say 1000 times the size—moving the thumb of the scroll-bar can produce jumps, and some positions may be unreachable. Therefore, scrolling alone is insufficient for navigating large spaces, hence the prevalence of the pan-and-zoom navigation method

Zooming and panning has been used since the beginning of interactive computer graphics to navigate in maps and other graphic scenes. Perlin introduced zoomable user interfaces in [24], while Furnas and Bederson have formalized the concept of space-scale diagrams to reason about these zoomable spaces.

Navigation involves not only viewing the space, but also moving the viewport. Several researchers have worked to optimize the coupling of zoom and pan to allow faster and more accurate navigation in zoomable interfaces [17, 11, 32, 27] with no assumptions regarding the contents of these spaces. Ishak and Feiner have proposed *Content-aware Scrolling* [19] to optimize navigation when paths are known in advance. For example, their system simplifies following the reading path of a 2-column document by interactively moving the viewport along the path, and dynamically adjusting the zoom to limit the rate of optical flow in a way similar to Speed-Dependent Automatic Zooming [17]. Our Link Sliding technique refines this idea for network navigation.

Navigation to Off-Screen Targets

More recent techniques have begun to address the problem of reaching known off-screen targets, both in terms of their visibility [12] and in navigating to them [18]. These techniques are closely related to our interest. Given n potential targets, some being off-screen, they provide visual indication of their location using a simple representation at the viewport's edge (either arcs or wedges). They then provide mechanisms to select items of interest, and to navigate to them quickly. These techniques are primarily aimed at small screens such as PDAs or smart phones, for reading maps, and reaching places of interest.

Other methods, such as the Vacuum [5], are designed for wall-sized displays, where targets may be too far away or too high to reach. These techniques rely on known targets

that are attracted using a directional probe; once they have been attracted their selection becomes possible. Instead of navigating to the items, the items are brought close to the pointer for examination and selection. These techniques are closely related to our Bring & Go technique but do not use any topological information to attract objects, only geometric information.

Space Distortion Techniques

Another approach to navigating large spaces is to distort the space to shrink uninteresting areas or magnify interesting ones. Magnifying lenses are the simplest of these techniques. They have been improved with fisheye lenses [23] that can present an overview of the space while allowing local in-context zooms, featuring a smooth transition between the two regions. Latest developments include Sigma Lenses [26], which use different dimensions to transition between the overview and the local zoom. However, the maximum zoom level is about 50, still limited compared to the size of spaces such as the surface of the Earth.

Other approaches include rubber-sheet deformations [30] and folded spaces [7] where parts of the space is folded, or stretched, to provide faster navigation with context awareness. These techniques address representation, and can cope with a variety of interaction techniques for deforming the space.

Dynamic Layout for Navigation

The above-mentioned techniques are based on a stable space that users want to explore. Network visualization systems start from a graph structure and compute a layout that creates the space. Changing the layout algorithm, or parameters of these algorithms, can change the space dramatically. The Bring Neighbors Lens [31] dynamically adjusts the graph layout to show local connectivity, but is not designed as a navigation technique. Some network visualization methods do use this possibility to facilitate navigation [34]. However, if not carefully constrained, such transformations may break the user's mental map of the network [21].

To a lesser extent, the geometry of links can be changed to enhance their legibility. EdgeLens [33] interactively distorts links around the pointer, to separate close links that are hard to follow visually, or that are simply overlaid. Conversely, Hierarchical Edge Bundles [15] group links to better show aggregated trends in graph connections, and can be tuned interactively. These simple operations do not change the layout, but improve the readability of links and can help users in navigating the network.

Topological Navigation

When a spatial embedding is automatically created from a graph, the size and visibility of the graph features can be interactively controlled by the user through a "degree of interest" function, introduced by Furnas [8]. Furnas considers that items in any topology can be assigned an *a priori* importance and an importance related to a focus point. When a user selects an item, he conveys to the system the information that this item is important to him. Usually, items in the neighborhood of the focused item are also important, but not

CHI 2009 ~ Understanding Graphs

April 9th, 2009 ~ Boston, MA, USA

as much so, and the importance decreases with some notion of topological distance. The representation of the topology should show the focus item and its neighborhood, and then allocate less screen real-estate and visibility to items that are farther away and less important.

Again, representation can be separated from interaction. Furnas describes the application of his framework to a tree structure based on selection. This method has been used effectively on trees [28] and on networks [9]. Issues with this approach include visual discontinuity when updating the degree of interest, and inconsistency in the user's mental map when the geometry is recomputed. Topological navigation allows arbitrarily large data structures to be navigated, as only a small subset is visible at any time. Navigation consists of successive selection of neighbor items or of textual search as in SpaceTree [28].

Current navigation techniques either ignore topological information to optimize spatial navigation, or consider mainly topology while attempting to maintain a consistent geometry. Our approach considers aspects of the geometry and topology at the same time.

AUGMENTING STANDARD NAVIGATION TECHNIQUES

Pan-and-zoom navigation is a standard technique that is used in a large number of 2D navigation tasks. It is an essential element of map and graph visualization software, and many similar applications. However, in the presence of numerous crossing paths, it can be difficult to follow a single path using this technique alone. Furthermore, if the distance to the destination is unknown, users may fail to zoom-out, and require numerous clutching operations to follow the path. Bird's eye views have been shown to be effective in navigating large 2D information spaces [16, 20, 27], but they do not allow the user to clearly resolve individual paths due to the large scaling factor and the small amount of screen real estate allotted to the view (Figure 1-a).

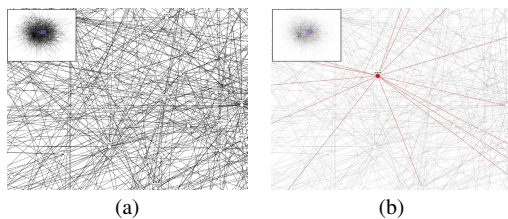


Figure 1. (a) Following a link in a dense graph can be difficult using standard Bird's Eye View navigation. (b) Highlighting a node's outgoing links make the task significantly easier.

We can simplify tasks that use connectivity information by visually distinguishing paths of interest from the background clutter. The techniques we implemented for the experiment below allow users to select a node of interest by clicking on it. The outgoing links are then colored in red, while the contrast of all other links, nodes, and labels (black by default) is reduced by rendering them a light-gray color (Figure 1(b)). Users may restore the default color-scheme by clicking again

on the initial node, or may select a different node thereby highlighting its exiting links.

Our Pan & Zoom implementation supports panning by allowing the user to grab-and-drag the graph by clicking on any non-node location. The motion of the graph follows that of the mouse cursor. We use the mouse wheel to zoom by doubling or halving the view scale at each click of the wheel. As our graph labels are illegible at scales sufficient for context, we have selected this mapping to facilitate rapid switching between wide and focused views. Moving the wheel forward zooms in, and backward zooms out. Our Bird's Eye View implementation allows users to center the view over any part of the graph by simply clicking on the corresponding location in the Bird's Eye View. Users may also pan the view as in the Pan & Zoom technique, or by dragging the viewport indicator rectangle in the Bird's Eye View. For the purpose of the study, our Bird's Eye View technique does not support zooming.

LINK SLIDING

Following a route on a map, or a link in a graph visualization, is essentially a one-dimensional navigation task. However, traditional navigation techniques, such as Pan & Zoom, require the control of two or three degrees-of-freedom to accomplish the task effectively. The Link Sliding technique simplifies the control task by constraining motion to a single path. The user slides a link-cursor along the link towards the destination node, as though sliding a bead on a wire. Changes in the direction of mouse movements are only necessary if the path curves sharply. Otherwise, the user may slide between two nodes by simply moving the mouse along the direction tangent to the path. This motion does not require a high degree of precision, as any movement perpendicular to the path is ignored, and motion stops at the destination node. The view is automatically panned to follow the mouse cursor, keeping it in its initial screen location, and the zoom level is adjusted so as to provide the user with additional context while sliding along the link.

To activate Link Sliding, the user simply presses and holds the mouse button on the start node. A light-gray circle appears around the node, indicating the selection-radius (Figure 2-a). This radius specifies a region of unconstrained mouse movement, which allows the user to select an outgoing link. A link-cursor is projected onto the closest point on the nearest link to aid in selection. As the cursor passes the selection-radius, the mouse cursor is drawn towards the link-cursor. Constrained sliding proceeds as follows: at each mouse event, the system updates the mouse-cursor position $\mathbf{p}$, and calculates $\mathbf{c}$, the closest point on the path to $\mathbf{p}$, assigning it to the path-cursor position. The mouse position is pulled towards $\mathbf{c}$, by setting $\mathbf{p}' = \alpha\mathbf{p} + (1 - \alpha)\mathbf{c}$. We set $\alpha = 1$ within the selection-radius to allow free mouse movement, and smoothly blend it towards 0 beyond the radius by updating it at each mouse motion event: $\alpha' = \beta\alpha + \omega$. Our system sets $\beta = 0.5$ and $\omega = 0$ to rapidly pull the mouse cursor to the link.

CHI 2009 ~ Understanding Graphs

April 9th, 2009 ~ Boston, MA, USA

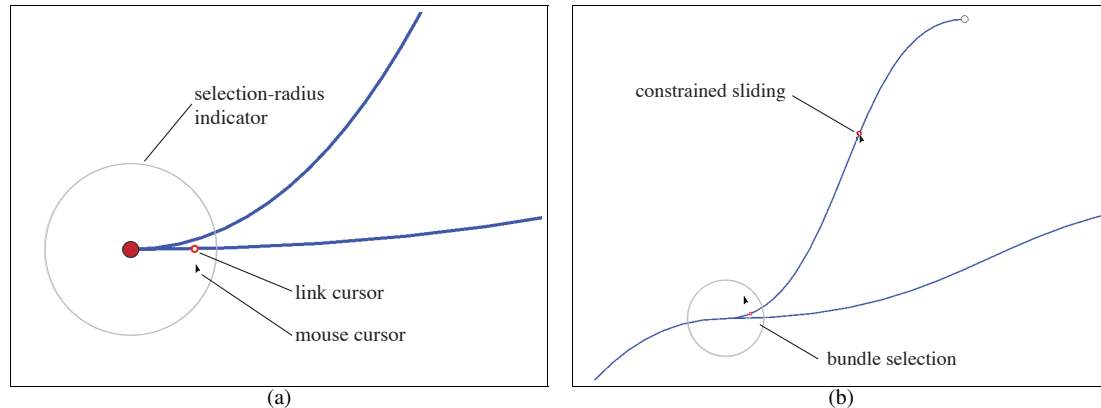


Figure 2. (a) The mouse cursor is free to move within a node's selection-radius. The link cursor shows the closest link, which will be selected upon passing the selection-radius. (b) Link selection can also be performed at junctions of edge-bundles, beyond a node or junction's selection-radius the mouse cursor is constrained to slide along the link.

Distance-dependent Automatic Zooming

Automatic zooming has been shown to reduce completion times in document navigation tasks [17]. However, previous work on speed-dependent automatic zooming [17] relied on rate-control panning and scrolling using a mouse. Pairing first-order control with an isotonic input device like the mouse is known to yield poor performance [35]. We introduce a zero-order control mapping that automatically zooms based on the user's position along a path. Distance-dependent Automatic Zooming (DDAZ) is possible, when both the start and end positions along a path are known (e.g. when following a route map). While a variety of useful zoom-level mappings are possible [19], for a graph navigation task we designed a mapping that sets the zoom level at the halfway point along the path so that the length of the path is equal to the viewport width. Thus, short links that traverse less than one screen-width will produce no zooming, while for long links, most of the remaining distance to the target is likely to be visible at the halfway point. Beyond providing the user with additional context, zooming the view adjusts the motor-space mapping to document space, allowing the user to move faster at the central part of the link. We use the following mapping (similar to a connes function): Given a distance d traveled by a user along a path of length l , the scale of the viewport is $z = (1 - (1 - (2d/l - 1)^6)^4) \times (1 - l/w)$, where w is the width of the viewport at $z = 1$. This mapping smoothly and rapidly reaches a wide view of the graph.

Sliding Along Edge Bundles

When a vertex in a graph has a large number of neighbors, it is helpful to gather the exiting links in *edge bundles* to reduce visual clutter, and aid in link selection [15]. Link Sliding has been designed to easily traverse edge bundles. Sliding along a bundle of links is identical to sliding along a single link until a junction in the bundle is reached. Around each bundle junction, a light-gray circle indicates a selection-radius where the mouse cursor is detached from the link cursor, al-

lowing the user to select an exiting link in the same manner as is done at the start node. At each mouse event, the closest point on the entire bundle is computed, and its position assigned to the link cursor. This allows the user to jump between nearby links in a bundle by moving the mouse cursor rapidly away from the current link to which it is constrained. Isolated links strongly maintain the sliding constraint, as no other link can be reached within a single mouse motion event. The strength of the mouse cursor's attraction to the bundle can be modified by adjusting the smoothing parameters of α as described above.

Bundling

When a node has many outgoing links, selecting one specific link becomes difficult. To overcome that problem, we aggregate all the links that point in the same direction and construct edge bundles. We limit the number of links starting from a node to k (3 to 5 depending on the user's preference). Therefore, bundles have to split at special positions we call "junction nodes". Our algorithm constructs bundles and junction nodes until the number of outgoing links from n is less than or equal to k .

The bundling algorithm consists of two stages. It first selects the links that are to be bundled, i.e.: 1) the link to the most distant neighbor of n , called n_f , and 2) the links to the nodes closest to n_f among the neighbors of n . It then adds a single link (the bundle root) from n to a created junction node n_j . The junction node is positioned at the center of the wedge formed by the bundled links, at a distance from n that is a fraction of the distance to the closest node of the bundle. The second stage consists of changing the bundled links' sources from n to n_j . This process is repeated until all the long links have been bundled.

Bundling is only performed when the user starts the navigation by selecting the source node. It is considered a navigation mode and not a rendering style as in [15].

CHI 2009 ~ Understanding Graphs

April 9th, 2009 ~ Boston, MA, USA

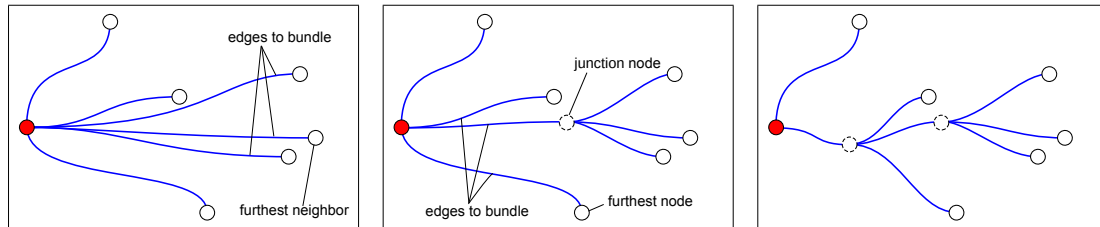


Figure 3. Multiple links leaving a node in the same compass direction (a) are collected into bundles by routing them through a dummy junction node (b). The process is repeated until the number of bundles leaving all nodes or junctions in the same compass direction fall below a given threshold (c).

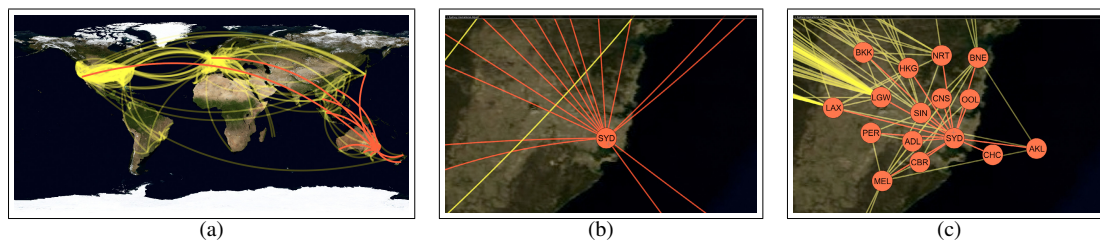


Figure 4. (a) Highlighting all flights to/from Sydney, Australia. (b) Close-up on Sydney, with highlighting. (c) Bring & Go initiated on Sydney.

BRING & GO

Link Sliding makes it easy to navigate along a given path. However, it does not help in the decision process that leads to the selection of one path among many potential candidates. This decision might depend on the type of arc to be followed when there are different types of paths. It might also depend on attributes of the node at the other end of the path. Having to navigate to the other end, in order to decide whether this is the path of interest or not, quickly becomes tedious as the number of connected arcs increases.

Bring & Go addresses this problem by bringing adjacent nodes close to a node upon selection. Figure 4-a shows a map of about 700 commercial flights connecting 232 airports. Highlighting (in red) gives a general idea of the number and location of airports connected to the currently selected node: Sydney International. At this scale, the node is difficult to select, being only 1-pixel large on a 17" display. Moreover, some parts of the network are very crowded, making it difficult to visually follow the paths. One has to zoom-in to get detailed information such as airport names, thus losing context and moving all airports connected to Sydney out of the viewport (Figure 4-b). When selecting the node corresponding to Sydney, Bring & Go translates all airports connected to it inside the current viewport (Figure 4-c) using smooth animations to preserve perceptual continuity [29]. The spline curves that represent links are smoothly attenuated and brought inside the viewport, thus providing additional contextual information, such as the degree of connected nodes, that might help the user make his decision. For instance, the user might be looking firstly for an airport hub, which would be more likely to offer him a direct flight to his final destination.

Once the user has made a decision about what link to follow, Bring & Go makes it very easy to reach the corresponding node with a simple selection of that node. The viewport is smoothly animated, zooming out and then in to get some context, as when traveling along a path with Link Sliding (see section on distance-dependent automatic zooming for details about the computation of the trajectory in space and scale). In the meantime, all nodes and splines are animated back to their previous position and geometry, thus restoring the network to its original configuration and terminating the Bring & Go.

The concept of Bring & Go is similar to Hopping [18] and Drag-and-Pop[3], which facilitate selection by bringing proxies of potential targets closer to the cursor. Bring & Go, however, is designed for navigation, rather than selection, and can handle a much larger number of distant targets, as only connected nodes are brought close. The technique is also similar to the Bring Neighbors Lens [31], which adjusts the layout of the graph to bring connected nodes into view. However, we believe that Bring & Go's use of proxies, rather than distortion, better preserves spatial constancy, and is critical for geographically embedded networks. Most importantly, Bring & Go works iteratively: a new Bring & Go can be initiated on a node that is currently inside the viewport as the result of a previous Bring & Go, bringing additional nodes into view, and so on. Looking for a flight from Sydney to Tel Aviv (which are not directly connected in our network), a user would easily identify London as a hub and, thanks to a second Bring & Go initiated on the London node brought inside the viewport, find that it is connected to Tel Aviv.

CHI 2009 ~ Understanding Graphs

April 9th, 2009 ~ Boston, MA, USA

Bring & Go Layout

The layout algorithm for computing the position of nodes brought inside the viewport is relatively simple. As illustrated in Figure 5, nodes are placed in concentric circles centered on the selected node according to the following method.

Polar coordinates (ρ , θ) are computed for all connected nodes. The list of nodes to be brought inside the viewport is sorted by distance ρ (shortest first). Nodes are then brought onto the rings following this order. For each node the algorithm checks each ring, starting from the innermost one, until it finds one where the node can be laid out, without overlapping any previously laid-out node, while keeping its θ coordinate constant. This method preserves the direction to the original location of brought nodes, and gives a sense of their relative distance to the selected node.

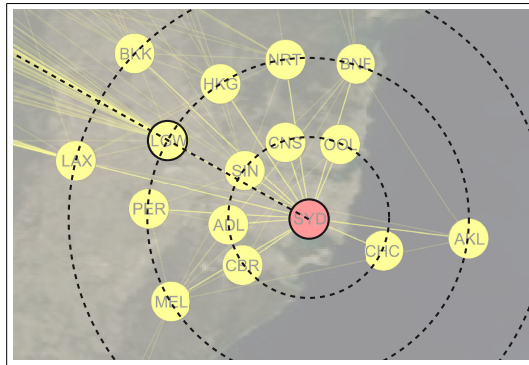


Figure 5. The layout of brought nodes preserves the direction to their actual location, and gives a sense of their relative distance to the selected node.

EXPERIMENT

We conducted an experiment to compare the performance and limits of Bring & Go (hereafter abbreviated B&G) and Link Sliding (LS) with simple visual augmentation of the methods currently available for navigating in node-link diagrams: Pan & Zoom (PZ), and interactive Bird's Eye View (BEV). Participants were asked to perform a compound navigation task on an abstract graph. We measured the performance time for each sub-task, and accuracy where relevant. Following the task participants responded to a questionnaire regarding their experience.

Task and Stimuli

All navigation tasks are set in one of two randomly generated scale-free graphs, one *sparse*, and one *dense*. The graphs were created using a Barabási-Albert model [2]. In the graph generation process, each new node is connected to n distinct nodes, randomly chosen from the existing nodes. The *sparse graph* (1000 nodes, 1485 edges) was generated with a random connectivity $n \in [1, 2]$, and the *dense graph* (1000 nodes, 2488 edges) using $n \in [1, 5]$. We also provide a small graph (200 nodes, 477 edges) for training purposes.

The nodes on the graphs are labeled with the names of animals, and vegetables. Unlike a real-world task, where the node labels are meaningful to the user, random name labels can be difficult to remember. To reduce the cognitive load on our participants we consistently give the start node the label "rat", and give one of its neighbors the label "cat" (explained below).

The trials in each condition are assigned to four fixed patterns in the corresponding graph, each consisting of a starting node and its neighbors. A pattern has a starting node of degree d ($d = 5$ in the *sparse* graph, and $d = 10$ in the *dense* graph). For each technique, participants perform timed tasks for all patterns of the two graphs. As we have four techniques, we provided four versions of each graph: the initial one, and its rotations by 90, 180 and 260 degrees. The labels are changed for each graph pattern, but retain their animal/vegetable category.

As each technique we study may be best suited to a different graph navigation tasks, we select three representative task from Lee *et al.*'s task taxonomy for graph visualization [22]. The tasks include identifying all nodes connected to a given node, following a link, and returning to a previously visited link. Participants performed the three tasks in sequence to complete one trial. The first task (*neighbors* task) is to identify a node's immediate neighbors. Participants begin at the start node, which is highlighted in orange, and are asked to count how many of the node's neighbors are labeled with an animal name, and to remember the location of the cat node. Participants press the space-bar to start, and press it again when they are done counting. After typing in the number of animal nodes, the system informs the participant if they have counted correctly. In the second task (*follow* task), the system centers the view about the start node, and participants are asked to follow a link, highlighted in orange, from the start node to a selected "visit node". Pressing the space-bar begins the task and clicking on the visit node completes it. Participants are then instructed to begin the third task (*revisit* task). Participants press the space-bar to start, and must then locate the cat node that they have seen in the *neighbors* task, and click on it. They may do this either by retracing their steps back to the start node, or by moving directly to where they believe the cat node is located.

To control the tasks completion time, the sum of the distances between the starting node and its neighbors is constant in all trials of the *sparse* graph, and all trials of the *dense* graph. The distance between the starting node and the cat node is constant in all patterns, as is the distance between the starting node and the visit node. Participants were given a maximum of 40 seconds to perform each task.

Design

Our experiment follows a 4×2 within-subject design: each participant performs tasks using each of the four techniques (*Technique* = PZ, BEV, B&G, LS) under two different graph conditions (*Graph* = *Sparse*, *Dense*). We group trials into four blocks, one per technique, so as not to confuse participants with frequent changes of the technique. To avoid

CHI 2009 ~ Understanding Graphs

April 9th, 2009 ~ Boston, MA, USA

ordering effects, we balance the presentation of technique using a Latin square consisting of four presentation orders, and randomly assign three participants per order. Within a *Technique* block, each participant perform eight measured trials, i.e., four trials with each of the two graphs. Trials within a block were presented in a random order after a training phase containing four trials, allowing participants to get familiar with a given technique before empirical measures were collected. Each navigation task (*neighbors*, *follow*, and *revisit*) within a trial had to be performed without any pause, but participants were allowed to rest between tasks. The experimenter first introduced the task, and then described each technique immediately before the corresponding block, to ensure that participants understood how each technique worked and how best to operate it.

Thirty-two measured trials per participant were analyzed, yielding a total of 384 trials:

$$\begin{array}{r} 12 \text{ Participants} \\ \times 4 \text{ Techniques} \\ \times 2 \text{ Graphs} \\ \times 4 \text{ Trials} \\ \hline 384 \text{ Total Trials} \end{array}$$

Apparatus

The experiment was ran on a HP Compaq 8710p equipped with a 2.4 GHz Intel Core 2 Duo processor, an Nvidia Quadro NVS 320M graphics card, a 1440 x 900 17" (43.2 cm) LCD monitor, and a Dell optical mouse. The mouse wheel produced 24 discrete clicks per revolution. The software was written in Java 1.6 using the Piccolo toolkit [4].

Participants

Twelve unpaid adult volunteers (8 male, 4 female), with ages ranging from 23 to 35 years, participated in the experiment. Participants were right-handed, with normal or corrected-to-normal vision.

Predictions*Neighbors Task*

H0 - Time performance rank: B&G will be the fastest for both graph densities, as all of the neighbors can be seen on the screen at once with legible labels. B&G will be followed by LS as it should make navigating to neighbors easy, and by BEV, which allows fast motion through the graph. PZ is expected to be the worst performer.

H1 - Density influence on LS: LS may be affected by graph density as participants must make more decisions while sliding along a bundle. This may cause its performance to drop below that of BEV for *dense graphs*.

Follow Task

H2 - Time performance rank: B&G will be the fastest for both graph densities, followed by LS; BEV and PZ will perform similarly, as following a single link requires a high degree of precision, which may be difficult to achieve using BEV.

H3 - Density influence on LS: We expect LS to be affected by graph density, possibly performing slower than BEV for *dense graphs*.

Revisit Task

H4 - Time performance rank: B&G will be the fastest for both graph densities, followed closely by LS which may support greater use of spatial memory; BEV and PZ are expected to be slower due to more difficult control, but as both support use of spatial memory we do not expect the difference to be as great as in the previous tasks.

Results

In the analysis reported below the performance times of trials that were not completed successfully are replaced by the mean time of successful trials for each condition.

Neighbors Task

Error Rate: An error in this task indicates that the participant's count of neighboring nodes labeled with animal names was incorrect. Of all errors 19% were committed after the participant reached the time limit. The rest were committed within the allotted time for the task.

A Friedman's α^2 test showed a significant effect of *Techniques* (Figure 6-a). Wilcoxon's signed ranks test revealed a significant difference for *dense graphs*. BEV (42% error rate) and LS (33%) were both significantly more error-prone than B&G (10%), while PZ showed a 25% error rate.

Performance Time: Results for this task should be interpreted cautiously on account of the high rate of errors. An analysis of variance (ANOVA) revealed a significant effect of *Techniques* on Time ($F_{3,33} = 69.811, p < .0001$) (Figure 6-b). Post-hoc pairwise mean comparisons showed that B&G was significantly faster than the three other techniques and that BEV performed better than PZ. Mean times were: PZ 27.2 sec (SD=0.8), BEV 21 sec (SD=0.7), LS 24 sec (SD=0.8) and B&G 8.5 sec (SD=0.2). ANOVA also revealed a significant effect of *Graphs* ($F_{1,11} = 242.587, p < .0001$) and a significant interaction *Techniques* $\times$ *Graphs* ($F_{3,33} = 21.82, p < .0001$). The performances of all *Techniques* were degraded with *dense graphs*. As predicted in *H0*, B&G showed fastest performance at visiting the neighborhood. LS, however was slower than expected. Moreover, the density of the graphs had an influence on time performance for all *Techniques*, whereas *H1* predicted that LS to be the most affected.

Follow Task

Error Rate: An error is reported when a participant was unable to complete the task within the time limit. Only two errors occurred, both for PZ in a dense graph. Both were due to subjects getting lost.

Performance Time: ANOVA revealed a significant effect of *Techniques* on Time ($F_{3,33} = 12.521, p < .0001$) (Figure 6(c)). Post-hoc pairwise mean comparisons showed that B&G was significantly faster than PZ and BEV and that LS was significantly faster than PZ. Mean times were: PZ 7 sec

CHI 2009 ~ Understanding Graphs

April 9th, 2009 ~ Boston, MA, USA

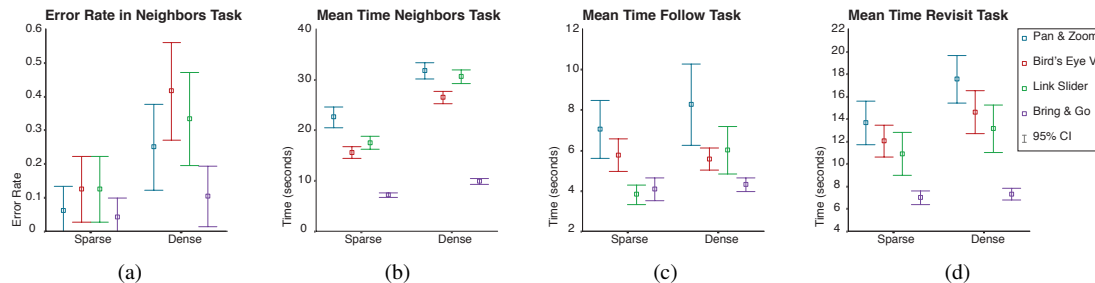


Figure 6. (a) Neighbours Task Error Rate and (b) Performance Time, (c) Follow Task Performance Time, (d) Revisit Task Performance Time.

(SD=0.4), BEV 5.6 sec (SD=0.2), LS 4.9 sec (SD=0.3) and B&G 4.2 sec (SD=0.2). ANOVA also revealed a significant effect of *Graphs* ($F_{1,11} = 5.81, p < .05$) and a significant interaction *Techniques* $\times$ *Graphs* ($F_{3,33} = 3.71, p < .05$). While the performances of PZ, BEV and B&G remains stable for both graphs densities, the performances of LS were degraded, with a mean time of 3.8 sec (SD=0.2) for *sparse graphs* and 6 sec (SD=0.6) for *dense graphs*. Here, our prediction for both time performance (H2) and graph density in uence on LS (H3) were verified.

Revisit Task

Error Rate: Nine errors occurred, 3 for PZ, BEV and LS, mainly for the dense graphs, where subjects did not find the cat node and time ran out.

Performance Time: ANOVA revealed a significant effect of *Techniques* on Time ($F_{3,33} = 35.453, p < .0001$) (Figure 6(d)). Post-hoc pairwise mean comparisons showed that B&G was significantly faster than the three other techniques and that LS was significantly faster than PZ. Mean times were: PZ 15.6 sec (SD=0.7), BEV 13.3 sec (SD=0.6), LS 12 sec (SD=0.7) and B&G 7.2 sec (SD=0.2). ANOVA also revealed a significant effect of *Graphs* ($F_{1,11} = 15.612, p < .01$). Performances on the *dense graphs* were significantly degraded. However, ANOVA did not show any significant interaction *Techniques* $\times$ *Graphs*. While our predictions on the time performance rank were verified, the difference between the spatial techniques and Bring & Go were greater than expected (H4).

Qualitative Evaluation

Following the task, participants responded to a questionnaire regarding their experience. Questions were presented using a five point labeled Likert scale with labels ranging from "strongly disagree" to "strongly agree". For each technique, participants responded to statements stating that it was easy to learn, was tiring, and was pleasant to use, that they were able to accomplish the tasks quickly using the technique, and that they found it easy to accomplish the *neighbors* and *revisit* tasks using the technique. Response summaries are presented in Figure 7. Written and oral comments were also solicited.

Participants unanimously agreed that Bring & Go was quick, and made accomplishing the tasks easy. They also found

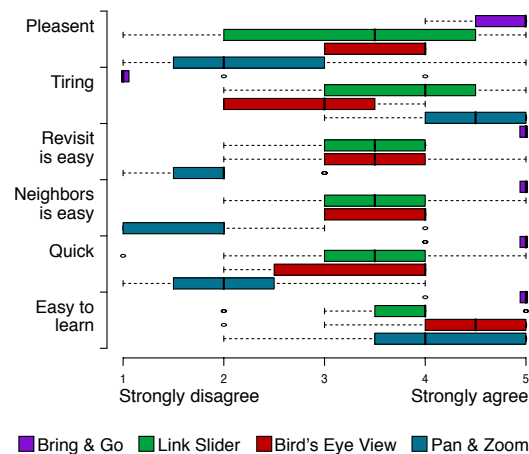


Figure 7. Questionnaire results for the four techniques.

it the least tiring, and most pleasant to use. Link Sliding and Bird's Eye View both received middling reactions regarding the ease and speed at accomplishing tasks. Pan & Zoom was generally rated as slow and difficult to use for the given tasks. A common complaint regarding Pan & Zoom was that the zoom control was too sensitive. We had set the zoom control to double or half the view scale at each click of the wheel in order to facilitate multi-scale navigation. This may be inconsistent with the wheel's more common use as a scrolling control, where a great amount of turning is needed to effect large changes.

Link Sliding elicited the most varied opinions regarding how pleasant it was to use. Some participants enjoyed the technique, one saying that it was "fun, and helped learning the graph layout," while others found it to be "tiring on the eyes." Several reported feeling dizzy or disoriented while sliding along links, attributing this feeling to the automatic zooming feature. The disorientation may also be due to rapid motion of the viewport while the participant's eyes were fixed on the cursor, which did not change screen location.

CHI 2009 ~ Understanding Graphs

April 9th, 2009 ~ Boston, MA, USA

DISCUSSION

Our experiment clearly illustrates that using connectivity information as a basis for graph navigation can significantly reduce task completion time, while improving the quality of the user experience. Specifically, the Bring & Go technique is faster and easier to use than the others we tested. While the other techniques do use connectivity information to some extent, they rely primarily on the spatial layout of the graph, and on the motion of the user's view port in this space. This is in sharp contrast with Bring & Go, which provides only slight spatial cues, and automatically controls the view port. The degree to which Bring & Go is faster is somewhat surprising, particularly for the *revisit* task, as one would expect its performance to improve with greater spatial awareness of the graph structure (*H4*). We believe that the ability to see all connected nodes quickly, and traverse links rapidly, greatly reduced our participants' need to rely on their memory.

One participant commented that Pan & Zoom "requires a lot of memory effort." This method does not allow a user to see all neighboring nodes at the same time while still being able to read their labels, consequently the user must remember the nodes' locations. Bring & Go makes remembering neighbor locations unnecessary, as they can all be seen at once. However this advantage may diminish in tasks that demand greater spatial awareness, such as navigating maps, or in networks where the location of a node is meaningful. Furthermore, the *revisit* task in our experiment only required users to traverse a single intermediate node with a known name. As the number of intermediate nodes increase, remembering a graphs connectivity may become more difficult than remembering its spatial embedding. Studying this trade-off is an interesting avenue for future research.

Beyond reducing memory requirements, Bring & Go also has a mechanical advantage over the other techniques, as the user does not need to pan the view. Panning is a closed-loop control task that must be performed almost continuously in the spatial navigation techniques, and which the user performs in parallel with a visual search. Bring & Go requires active control for only two pointing sub-tasks, and the visual search is performed separately. The difficulty of spatial navigation is reflected in participants comments on Pan & Zoom, as some note that they never used the zooming feature, finding it too difficult to control both position and scale at the same time. This difficulty may explain the high variance we observed in the Pan & Zoom technique, as some participants may have relied largely on zooming for navigation, while others would have relied more on panning.

Two key features of Link Sliding, that we expected would raise its performance over the other spatial techniques, are the simplification of the control task by reducing it to the control of a single degree of freedom, and the automatic control of the view port zoom level. Indeed, for the elementary task of following a single link, Link Sliding appeared to perform at least as well as Bring & Go. This can be seen in the *follow* task in the sparse graph condition (Figure 6-c). However, this performance did not scale favorably with the

complexity of the task (*H1*). Visiting all of the neighbors of a node requires repeated round-trips back to the start node. In contrast, using Bird's Eye View participants were able to move directly from neighbor to neighbor. A number of our participants commented that it was too difficult to jump between adjacent links, and one explicitly requested a mechanism for shortcuts between links. Another possible barrier to allowing Link Sliding to navigate easily from high-valence nodes is the effect of edge-bundling. While edge-bundling reduces clutter, and simplifies link selection when leaving a node, it also changes the visual orientation of links, and requires the user to follow a more complex path with multiple junctions that are not a part of the underlying graph's topology. The change in angle can be particularly problematic when the user attempts to return to the previous node after releasing the mouse button, as the new node's outgoing bundles do not match the original node's bundles, so the user must leave by following a bundle aimed in a slightly different direction than of the bundle on which she arrived. We believe that some of these issues may be addressed by optimizing the bundling algorithm to minimize changes in link angle, or by finding an optimal set of static edge-bundles for the entire graph.

CONCLUSION

This work has begun the exploration of topology-aware navigation of large graphs, by examining several possible methods ranging from mostly spatial, to mostly topology-based navigation. We have found the technique Bring & Go, which relies more on topology than spatial location, to hold a clear advantage for several key navigation tasks. The Link Sliding technique, which attempts to combine the advantages of both spatial and topological navigation, did not perform as well as expected, performing the same as, or only slightly better than Bird's Eye View. However, we believe that Link Sliding is worthy of further investigation for navigating networks, such as route-maps, that are spatially embedded. While Bring & Go works well for finding labeled nodes, it provides very little geographical context. For example, finding all of the coastal towns of an unknown country would be a difficult task using Bring & Go, as the user cannot easily follow the shoreline. This task could be easily accomplished using either Link Sliding or Bird's Eye View navigation.

Our results suggest that any system for visualizing large networks will profit from some form of topology-aware navigation using one or more of the techniques described here. As both the visual augmentation, as well as the navigation techniques, are triggered only in the context of links and nodes, they do not interfere with existing spatial navigation. Indeed, a combination of methods may be required for high-level navigation tasks, as each method has its own unique strengths.

Bring & Go and Link Sliding have been implemented using both Piccolo [4] and ZVTM [25], and are available in ZGRViewer¹, a tool for navigating large graph layouts computed using the GraphViz package².

¹<http://zvtm.sf.net/zgrviewer.html>

²<http://www.graphviz.org/>

CHI 2009 ~ Understanding Graphs

April 9th, 2009 ~ Boston, MA, USA

REFERENCES

1. E. Adar. Guess: a language and interface for graph exploration. In *CHI 06*, 791–800, New York, NY, USA, 2006. ACM.
2. A. L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.
3. P. Baudisch, E. Cutrell, D. Robbins, M. Czerwinski, P. Tandler, B. Bederson, and A. Zierlinger. Drag-and-pop and drag-and-pick: Techniques for accessing remote screen content on touch and pen-operated systems. In *Proc. Interact 03*, 57–64, 2003.
4. B. B. Bederson, J. Grosjean, and J. Meyer. Toolkit design for interactive structured graphics. *IEEE Trans. on Software Engineering*, 30(8):535–546, 2004.
5. A. Bezerianos and R. Balakrishnan. The vacuum: facilitating the manipulation of distant objects. In *CHI 05*, 361–370, New York, NY, USA, 2005. ACM.
6. Delta Air Lines. Delta air lines route map, 2008. <http://delta.innosked.com/>.
7. N. Elmqvist, N. Henry, Y. Riche, and J.-D. Fekete. Melange: space folding for multi-focus interaction. In *CHI 08*, 1333–1342, New York, NY, USA, 2008. ACM.
8. G. W. Furnas. Generalized fisheye views. *SIGCHI Bull.*, 17(4):16–23, 1986.
9. E. R. Gansner. Topological fisheye views for visualizing large graphs. *IEEE Trans. on Visualization and Computer Graphics*, 11(4):457–468, 2005.
10. Google. Google maps, 2008. <http://maps.google.com>.
11. Y. Guiard, F. Bourgeois, D. Mottet, and M. Beaudouin-Lafon. Beyond the 10-bit barrier: Fitts' law in multi-scale electronic worlds. In *HCI 01*, 573–588. Springer, 2001.
12. S. Gustafson, P. Baudisch, C. Gutwin, and P. Irani. Wedge: clutter-free visualization of off-screen locations. In *CHI 08*, 787–796, New York, NY, USA, 2008. ACM.
13. J. Heer, S. K. Card, and J. A. Landay. prefuse: a toolkit for interactive information visualization. In *CHI 05*, 421–430, New York, NY, USA, 2005. ACM.
14. I. Herman, G. Melancon, and M. S. Marshall. Graph visualization and navigation in information visualization: A survey. *IEEE Trans. on Visualization and Computer Graphics*, 06(1):24–43, 2000.
15. D. Holten. Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *IEEE Trans. on Visualization and Computer Graphics*, 12(5):741–748, 2006.
16. K. Hornbæk, B. B. Bederson, and C. Plaisant. Navigation patterns and usability of zoomable user interfaces with and without an overview. *ACM Trans. Comput.-Hum. Interact.*, 9(4):362–389, 2002.
17. T. Igarashi and K. Hinckley. Speed-dependent automatic zooming for browsing large documents. In *UIST 00*, 139–148, New York, NY, USA, 2000. ACM.
18. P. Irani, C. Gutwin, and X. D. Yang. Improving selection of off-screen targets with hopping. In *CHI 06*, 299–308, New York, NY, USA, 2006. ACM.
19. E. W. Ishak and S. K. Feiner. Content-aware scrolling. In *UIST 06*, 155–158, New York, NY, USA, 2006. ACM.
20. V. Kaptelinin. A comparison of four navigation techniques in a 2d browsing task. In *CHI 95*, 282–283, New York, NY, USA, 1995. ACM.
21. H. Lam, R. A. Rensink, and T. Munzner. Effects of 2d geometric transformations on visual memory. In *APGV 06*, 119–126, New York, NY, USA, 2006. ACM.
22. B. Lee, C. Plaisant, C. S. Parr, J.-D. Fekete, and N. Henry. Task taxonomy for graph visualization. In *BE-LIV 06*, 1–5, New York, NY, USA, 2006. ACM.
23. Y. K. Leung and M. D. Apperley. A review and taxonomy of distortion-oriented presentation techniques. *ACM Trans. Comput.-Hum. Interact.*, 1(2):126–160, 1994.
24. K. Perlin and D. Fox. Pad: an alternative approach to the computer interface. In *SIGGRAPH 93*, 57–64, New York, NY, USA, 1993. ACM.
25. E. Pietriga. A toolkit for addressing hci issues in visual language environments. In *IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC 05)*, 145–152, Los Alamitos, CA, USA, 2005. IEEE Computer Society.
26. E. Pietriga and C. Appert. Sigma lenses: focus-context transitions combining space, time and translucence. In *CHI 08: Proceeding of the twenty-sixth annual CHI conference on Human factors in computing systems*, 1343–1352, New York, NY, USA, 2008. ACM.
27. E. Pietriga, C. Appert, and M. Beaudouin-Lafon. Pointing and beyond: an operationalization and preliminary evaluation of multi-scale searching. In *CHI 07*, 1215–1224, New York, NY, USA, 2007. ACM Press.
28. C. Plaisant, J. Grosjean, and B. B. Bederson. Spacetree: Supporting exploration in large node link tree, design evolution and empirical evaluation. In *INFOVIS 02*, 57, Washington, DC, USA, 2002. IEEE Computer Society.
29. G. G. Robertson, S. K. Card, and J. D. Mackinlay. Information visualization using 3d interactive animation. *Communication of the ACM*, 36(4):56–71, 1993.
30. M. Sarkar, S. S. Snibbe, O. J. Tversky, and S. P. Reiss. Stretching the rubber sheet: a metaphor for viewing large layouts on small screens. In *UIST 93*, 81–91, New York, NY, USA, 1993. ACM.
31. C. Tominski, J. Abello, F. van Ham, and H. Schumann. Fisheye tree views and lenses for graph visualization. In *IV 06: Proceedings of the conference on Information Visualization*, 17–24, Washington, DC, USA, 2006. IEEE Computer Society.
32. J. J. van Wijk and W. A. Nuij. A model for smooth viewing and navigation of large 2d information spaces. *IEEE Trans. on Visualization and Computer Graphics*, 10(4):447–458, 2004.
33. N. Wong, S. Carpendale, and S. Greenberg. Edgelens: An interactive method for managing edge congestion in graphs. In *InfoVis 03*, 51–58. IEEE Computer Society, 2003.
34. K.-P. Yee, D. Fisher, R. Dhamija, and M. Hearst. Animated exploration of dynamic graphs with radial layout. In *INFOVIS 01*, 43, Washington, DC, USA, 2001. IEEE Computer Society.
35. S. Zhai. User performance in relation to 3d input device design. *SIGGRAPH Comput. Graph.*, 32(4):50–54, 1998.

Representation-Independent In-Place Magnification with Sigma Lenses

Emmanuel Pietriga, Olivier Bau, and Caroline Appert

Abstract—Focus+context interaction techniques based on the metaphor of lenses are used to navigate and interact with objects in large information spaces. They provide in-place magnification of a region of the display without requiring users to zoom into the representation and consequently lose context. In order to avoid occlusion of its immediate surroundings, the magnified region is often integrated in the context using smooth transitions based on spatial distortion. Such lenses have been developed for various types of representations using techniques often tightly coupled with the underlying graphics framework. We describe a representation-independent solution that can be implemented with minimal effort in different graphics frameworks, ranging from 3D graphics to rich multiscale 2D graphics combining text, bitmaps, and vector graphics. Our solution is not limited to spatial distortion and provides a unified model that makes it possible to define new focus+context interaction techniques based on lenses whose transition is defined by a combination of dynamic displacement and compositing functions. We present the results of a series of user evaluations that show that one such new lens, the speed-coupled blending lens, significantly outperforms all others.

Index Terms—Graphical user interfaces, visualization techniques and methodologies, interaction techniques, evaluation/methodology.

1 INTRODUCTION

BIFOCAL display techniques complement conventional navigation techniques such as pan and zoom, typically used in 2D multiscale environments [1], [2], [3], and those based on the *walking* and *flying vehicle* [4] metaphors commonly employed to interactively explore 3D worlds. Beyond the now-ubiquitous map-like overviews introduced 30 years ago [5], a variety of bifocal techniques termed *focus+context* have been designed to further help users navigate in complex visual representations such as large trees [6], [7], graphs [8], and high-resolution bitmap representations [9]. These techniques can also help users interact with objects in vector graphics editors [10] or select small interface widgets [11]. The defining characteristic of these focus+context techniques is that they provide *in-place* magnification of a region (the focus) of the current display (the context), allowing users to get more detailed information about specific elements of the interface without having to zoom in the whole representation.

Interactive in-place magnification lenses have been available on the user's desktop for more than a decade, from simple magnifier lenses used as accessibility tools [12] to fancy screen savers distorting the user's workspace. These early implementations were based on a simple magnification method that consisted in merely duplicating the pixels of the original representation. While implementing such lenses is fairly trivial, implementing lenses that actually render objects in the magnified region with more

detail is not. Models and implementations of the latter usually require information about, and some level of control on, the inner structure and elements of the representation, which might actually get modified by the lens. Such models and implementations are thus often tightly coupled with a particular graphics framework or application. From a purely theoretical perspective, however, the same general approach applies to many types of applications: 2D vector graphics editors, geographical information systems, CAD tools, or any other 3D application. No matter the representation and underlying graphics framework, the process consists in rendering a subregion F of the current representation C at a larger scale and with more detail, and integrating F into C through a nonlinear transformation to achieve a smooth transition between the two.

The Sigma Lens framework [13] extends this general process by defining transitions between focus and context as a combination of dynamic displacement and compositing functions, making it possible to create a variety of lenses that use techniques other than spatial distortion to achieve smooth transitions between focus and context, and whose properties adapt to the users' actions, all in an effort to facilitate interaction. In this paper, we extend and strengthen that framework by describing a general approach to its design and implementation that shows how Sigma Lenses can be representation independent. The main contribution is a rendering technique based on a unified model that can be integrated with minimal effort in different graphics frameworks, ranging from 3D graphics consisting of complex textured meshes to rich multiscale 2D graphics combining text, bitmaps, and vector graphics. The technique does not need to have knowledge about, or access to, the graphical objects that constitute the representation.

After an overview of related work in Section 2, we give a theoretical description of our framework in Section 3, also summarizing the Sigma Lens unified model originally

- The authors are with INRIA Saclay—Ile-de-France and LRI (Université Paris-Sud & CNRS), Bat. 490, Université Paris-Sud, 91405 Orsay Cedex, France. E-mail: emmanuel.pietriga@inria.fr, {bau, appert}@lri.fr.

Manuscript received 18 Mar. 2009; revised 1 Aug. 2009; accepted 9 Aug. 2009; published online 18 Aug. 2009.

Recommended for acceptance by R. Balakrishnan.

For information on obtaining reprints of this article, please send e-mail to: tcvg@computer.org, and reference IEEECS Log Number TVCG-2009-03-0060. Digital Object Identifier no. 10.1109/TVCG.2009.98.

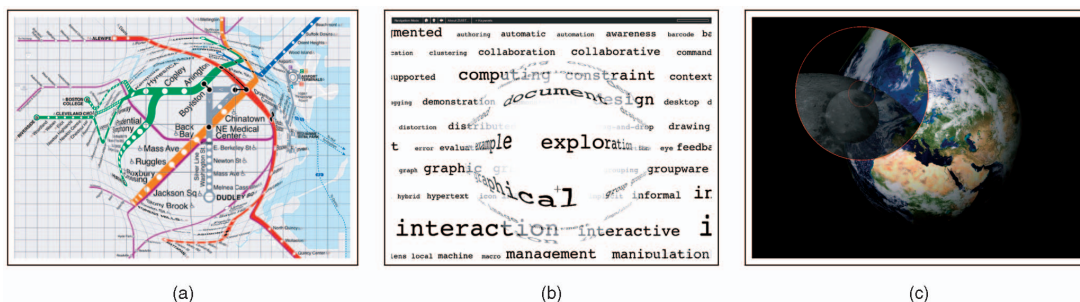


Fig. 1. Three different lenses obtained with minor modifications to the scale and compositing functions: (a) a distortion lens on a high-resolution bitmap (subway map), (b) a hovering lens on 2D vector graphics and text (ACM UIST keyword tag cloud), and (c) a speed-coupled blending lens on a 3D model of the Moon orbiting the Earth.

defined in [13]. We then instantiate lenses of particular interest in this model, and present a series of evaluations that test their limit performance on 2D navigations tasks. We provide a detailed summary of the experiments reported in [13], and report the results of a new controlled experiment that covers cognitive aspects of interaction with lenses that were left as future work, further confirming that one of our new lens designs, namely the SPEED-COUPLED BLENDING lens, significantly outperforms all others. A technical overview of two implementations follows, one for OpenGL 3D graphics taking advantage of hardware acceleration, the other for a general purpose, cross-platform 2D application programming interface. Advantages and limitations of our approach are discussed in Section 8, along with plans for future work.

2 RELATED WORK

Many focus+context techniques are based on the concept of lenses. The simplest lens is the electronic equivalent of a magnifying glass. While easy to understand, this type of lens occludes the immediate surroundings of the magnified region [14], as a physical magnifying glass does, thus hiding an area of significant interest and making it difficult for users to relate focus and context in the representation. In order to avoid this problem, the magnified region is often integrated in the context using smooth transitions based on nonlinear magnification techniques. These techniques create a transition zone in which the representation is distorted (see Fig. 1a).

Distortion-based visualization techniques often rely on metaphors inspired by the physical world such as stretchable rubber sheets [15] and, more generally, surface deformations [16]. Others work with more fundamental concepts such as hyperbolic projection [6] or nonlinear magnification fields [17]. The distortion can either extend to the limits of the representation [14], [15], or it can be bounded to a specific area. This paper focuses more specifically on the latter case, i.e., on lenses, termed *constrained lenses* [9], [17], [18], [19], that leave a significant part of the context undistorted. Cockburn et al. [20] survey many of these techniques in their recent review of overview+detail, zooming, and focus+context interfaces.

While all the above techniques apply to 2D graphics, other techniques have been developed for distortion in 3D.

A first set of techniques deforms 3D representations by projecting a texture on a mesh that models the distortion, as do pliable surfaces for 2D representations [16]. LaMar et al.'s magnification lenses [21] are based on homogeneous texture coordinates and special geometries. They can be applied to both 2D and 3D representations but are limited in the type of distortion and lens shapes they can model. Nonlinear perspective projections [22] project the RGB image produced by a 3D pipeline on a surface shape inserted in front of the flat projection plane. They can model spatially bounded distortions. Related to the latter is Brosz et al.'s single camera flexible projection framework [23], which is capable of modeling nonlinear projections through the parametric representation of the viewing volume.

There is also an impressive set of space deformation techniques (see [24] for an overview), ranging from early works on the deformation of solid primitives [25], [26] to view-dependent geometry [27] and deformation based on hardware-accelerated displacement mapping [28], [29] and deflectors [30]. These techniques distort 3D geometry, but often do so in an object-centric manner, and are thus not well suited to the implementation of focus+context navigation lenses, which deform a region of the current display, i.e., a subsection of the current viewing frustum that intersects a set of objects, some of them only partially. Camera textures [31] are among the few to actually apply constrained magnification lenses to 3D meshes, but the technique requires a sufficient level of tessellation of the target mesh to produce distortions of good quality.

Techniques such as the last one are typical examples of approaches strongly coupled with a specific type of graphics. This coupling is both a strength and a weakness. The strength lies in the capacity to access and modify the graphical objects that are to be distorted and rendered in a focus+context view. This is also a weakness, however, as a technique that works with 2D vector graphics will not be applicable to 3D meshes, or even to representations that include bitmap images. Other techniques, such as those based on 3D surface deformation and texture mapping, suffer from a somewhat opposite problem: while generating focus+context views of 2D representations, they require capabilities that are often not available in 2D graphics libraries.

One goal of our approach is to provide a method for implementing constrained magnification lenses that is as

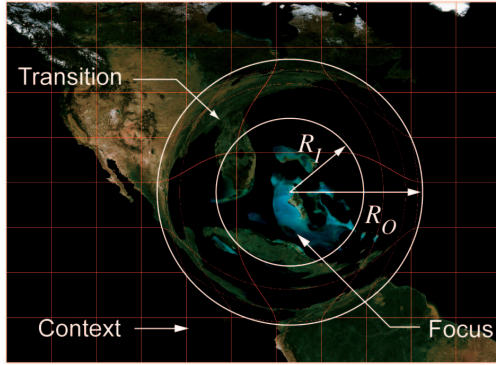


Fig. 2. Gaussian distortion lens. The level of detail in the flattop is increased by a factor of $MM = 4.0$.

independent as possible of the nature of the representation and graphics library employed, ranging from 3D scenes in OpenGL to rich 2D vector graphics, as typically manipulated with tools such as Adobe Illustrator. Our other goal is to create a flexible framework of higher expressive power than existing solutions, going beyond distortion to achieve smooth visual transitions between focus and context, resulting in lenses that can be manipulated more efficiently by users.

3 GENERAL FRAMEWORK

Our general approach is positioned at a level of abstraction high enough for the model to be applicable to a variety of graphics frameworks, requiring the underlying libraries to provide as small a number of features as possible. It basically consists in transforming the representation at the pixel level after it has been rendered, independently of how it was rendered. To employ terminology drawn from 3D graphics, our technique works in image space, as opposed to object space. This approach has advantages beyond its wide applicability but also some limitations, which will be discussed later in Section 8.

3.1 Dual Rendering

All constrained magnification lenses featuring a regular shape share the following general properties, no matter how they transition between focus and context (see Fig. 2):

- R_I : the radius of the focus region (a.k.a. the flattop), which we call *inner radius*,
- R_O : the radius of the lens at its base, i.e., its extent, which we call *outer radius*,
- MM : the magnification factor in the flattop.

Applying a constrained lens to a representation effectively splits the viewing window into two regions: the *context region*, which corresponds to the part of the representation that is not affected by the lens, and the *lens region*, in which the representation is transformed. We consider the lens and context regions as separate buffers: the *context buffer* holds what is displayed in the absence of any lens; the *lens buffer* contains a rendering of the region corresponding to the bounding box of the lens. These

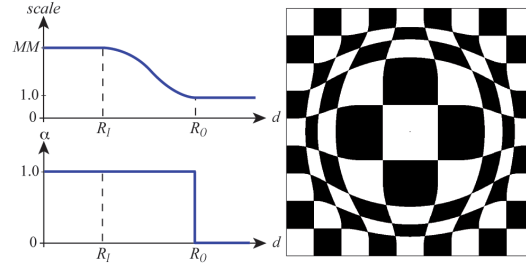


Fig. 3. Simple distortion lens.

basically correspond to the context and focus viewing windows of the original Sigma Lens model [13] in the case of 2D multiscale representations, though here we are not limited to this type of graphics representation.

Our technique consists in asking the underlying graphics library for two separate rendering passes. One corresponds to what is seen in the context region and is stored in the context buffer, whose dimensions $w \times h$ match that of the final viewing window displayed to the user. The other rendering pass corresponds to what is seen in the lens region. Since we want the lens to actually provide a more detailed representation of objects in the magnified region, and not merely duplicate pixels from the previous rendering, the actual dimensions of the lens buffer are $2 \cdot MM \cdot R_O \times 2 \cdot MM \cdot R_O$. The buffer can thus accommodate a uniform magnification by a factor of MM of the lens region. The final viewing window displayed on screen can then be obtained through the arbitrary transformation and composition of pixels from both buffers. This includes displacement and compositing functions that will control the transition between the focus and context regions, as defined in the Sigma Lens unified model [13], which we briefly summarize.

3.2 Sigma Lenses

Sigma lenses build upon prior work on distortion lenses [18], but combine space, time, and translucence to define new types of transitions between focus and context, in an effort to create more usable lenses (see Sections 5 and 6).

The standard transformation performed by graphical fish-eyes consists in displacing all points in the focus buffer to achieve a smooth transition between focus and context through spatial distortion. This type of transformation can be defined through a drop-off function which models the magnification profile of the lens. The drop-off function is defined as

$$G_{scale} : d \mapsto s,$$

where d is the distance from the center of the lens and s is a scaling factor. Distance d is obtained from an arbitrary distance function $\mathcal{D}$. A Gaussian-like profile is often used to define drop-off function G_{scale} , as it provides one of the smoothest visual transitions between focus and context (see Figs. 2 and 3). It can be replaced by other functions, which are already well-described in literature [9], [18].

Definition 1. Displacement and compositing function $\mathcal{R}$

$$\mathcal{R}(x, y) = \begin{cases} \left(x_c + \frac{x-x_c}{MM}, y_c + \frac{y-y_c}{MM} \right) \otimes_{\alpha_{FT}} (x, y), & \{\forall(x, y) | \mathcal{D}(x, y) \leq R_I\}, \\ \left(x_c + \frac{x-x_c}{\mathcal{G}_{scale}(\mathcal{D}(x, y))}, y_c + \frac{y-y_c}{\mathcal{G}_{scale}(\mathcal{D}(x, y))} \right) \otimes_{\mathcal{G}_{comp}(\mathcal{D}(x, y))} (x, y), & \{\forall(x, y) | R_I < \mathcal{D}(x, y) < R_O\}, \\ (x, y), & \{\forall(x, y) | \mathcal{D}(x, y) \geq R_O\}. \end{cases} \quad (1.1)$$

Distance functions producing basic regular lens shapes are easily obtained through $L(P)$ -metrics [18]:

$$\mathcal{D} : (x, y) \mapsto \sqrt[P]{|x - x_c|^P + |y - y_c|^P},$$

where (x, y) are the coordinates of a point seen through a lens centered in (x_c, y_c) , and $P \in \mathbb{N}^*$. $P = 2$ corresponds to a circular lens and $P = \infty$ to a square lens. As discussed later, more complex or irregular lens shapes can easily be obtained, e.g., by making R_I and R_O angle dependent, usually in combination with $L(2)$.

In our approach, the overall process consists in applying a displacement function to all pixels in the lens buffer that fall into the transition zone: pixels between R_I and $MM \cdot R_O$ get compressed according to the drop-off function in such a way that they eventually all fit between R_I and R_O . Pixels of the lens buffer can then be composited with those of the context buffer that fall into the lens region.

When only interested in spatial distortion, generating the final representation simply consists in replacing pixels in the lens region of the context buffer by those of the lens buffer, in other words, compositing them with the over operator ($\alpha = 1.0$). But, other values of α and other operators in Porter and Duff's rich algebra [32] can be used to achieve interesting visual effects. It is, for instance, possible to obtain smooth, distortion-free transitions between focus and context by applying an alpha blending gradient centered on the lens. Or, as we will see later, a simple magnifier lens ($R_I = R_O$) can be made much more usable by making it uniformly translucent and coupling α to its speed.

The rendering of a point (x, y) in the final viewing window is controlled by function $\mathcal{R}$ (see Definition 1), where

$$Plens \otimes_{\alpha} P_{context}$$

denotes the pixel resulting from alpha blending a pixel from the lens buffer and another from the context buffer with an alpha value of α . As with scale for distortion lenses, the alpha blending gradient can be defined by a drop-off function that maps a transluence level to a point (x, y) located at a distance d from the lens center:

$$\mathcal{G}_{comp} : d \mapsto \alpha,$$

where α is an alpha blending value in $[0, \alpha_{FT}]$, α_{FT} being the transluence level used in the flattop of the lens.

The flattop region corresponds to case (1.1) of Definition 1, the transition to case (1.2), and the region beyond the lens boundaries, i.e., the context, corresponds to case (1.3). Detailed information about the model and functions summarized in this section can be found in the original paper on Sigma Lenses [13].

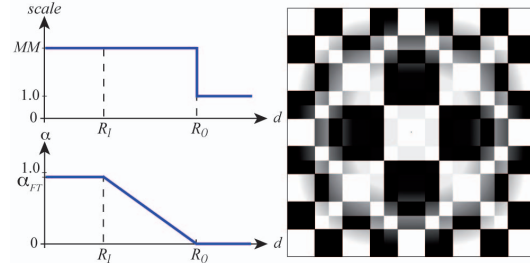


Fig. 4. Blending lens.

4 INSTANTIATING LENSES IN THE MODEL

In our approach to the implementation of the Sigma Lens framework, new lenses are obtained very easily, only by defining functions $\mathcal{G}_{scale}$ and $\mathcal{G}_{comp}$. Some examples of interesting transitions follow.

First, for reference, a plain and simple distortion lens with a Gaussian-like drop-off function (Fig. 3) can simply be obtained with the following instantiations of $\mathcal{G}_{scale}$ and $\mathcal{G}_{comp} : \{\forall d | R_I < d < R_O\}$

$$\begin{cases} \mathcal{G}_{scale}(d) = \frac{MM-1}{2} \cdot \cos\left(\frac{\pi}{R_O-R_I} \cdot d - \frac{\pi R_I}{R_O-R_I}\right) + \frac{MM+1}{2}, \\ \mathcal{G}_{comp}(d) = 1.0. \end{cases}$$

Figs. 1a and 2 give examples of a distortion lens with a Gaussian-like drop-off.

The BLENDING lens described in [13], which achieves a smooth transition between focus and context through gradual alpha blending only (Fig. 4), is easily instantiated using a linear drop-off for α :

$$\{\forall d | R_I < d < R_O\} \begin{cases} \mathcal{G}_{scale}(d) = MM, \\ \mathcal{G}_{comp}(d) = \frac{\alpha_{FT}}{R_I-R_O} \cdot d - \frac{\alpha_{FT} \cdot R_O}{R_I-R_O}. \end{cases}$$

This type of lens does not feature a continuous spatial transition between focus and context. Instead, visual continuity is achieved through increasing transluence.

As with any lens, other distance functions can be used [18]. $L(\infty)$, however, introduces artificial edges which give a false impression of spatial depth; if a square lens is required, $L(3)$ offers an interesting approximation, smoothing the edges while featuring a shape close to a square, as illustrated in Fig. 5.

Spatial distortion and gradual alpha blending can of course be combined. Fig. 6 shows an example of a so-called HOVERING lens as it appears to float above the representation.

$$\{\forall d | R_I < d < R_O\} \begin{cases} \mathcal{G}_{scale}(d) = \frac{1-MM}{R_O-R_I} \cdot d + \frac{MM \cdot R_O - R_I}{R_O-R_I}, \\ \mathcal{G}_{comp}(d) = \frac{\alpha_{FT}}{R_I-R_O} \cdot d - \frac{\alpha_{FT} \cdot R_O}{R_I-R_O}. \end{cases}$$

This lens mitigates issues of both distortion-only and alpha blending-only transitions: 1) when the user is performing a focus targeting action,¹ the target object no longer appears to temporarily move away from the

1. The low-level and ubiquitous action which consists in moving the lens in the main window so as to position it over an object to be magnified in the flattop is termed *focus targeting* [33].

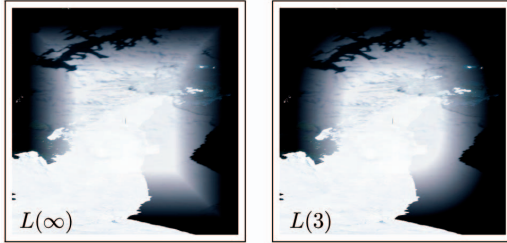


Fig. 5. Smoothing lens edges, on a magnified view of the Antarctic peninsula [34].

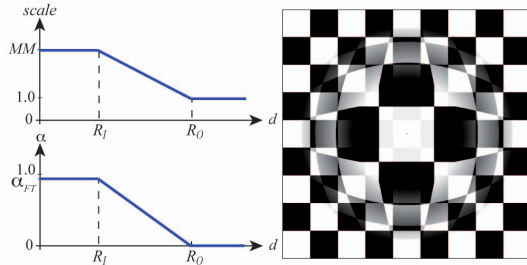


Fig. 6. Hovering Lens.

approaching flattop when entering the peripheral zone of the transition region (an effect due to the spatial distortion) as this zone is almost transparent; 2) this peripheral zone is still visible (undistorted) by transluence through the inner, distorted zone of the transition; 3) the distortion in this inner region contributes to visually differentiating focus and context during lens movements, and to the minimization of the distance between the point where an object disappears from the context and the point where it appears in the focus area.

4.1 Speed Coupling

In addition to the transition functions described earlier, the Sigma Lens framework allows for lens properties such as magnification factor, radius, or flattop opacity to vary over time. The first example of lens to make use of dynamic properties was Gutwin's SPEED-COUPLED FLATTENING lens [33], which uses the lens' dynamics (velocity and acceleration) to automatically control magnification. By canceling distortion during focus targeting, speed-coupled flattening lenses improve the usability of distortion lenses. Basically, MM decreases toward 1.0 as the speed of the lens (operated by the user) increases, therefore flattening the lens into the context, and increases back to its original value as the lens comes to a full stop. Such behavior can easily be implemented in our approach using a simple interpolated low-pass filter (see [13] for detailed information). Let $S(t)$ be the time-based function returning a numerical value that depends on the velocity and acceleration of the lens over time. The function is set to return a real value in $[0.0, 1.0]$. Making a lens parameter such as the magnification factor MM speed dependent is then easily achieved by simply multiplying that parameter by the value of $S(t)$, as shown in Fig. 7a.

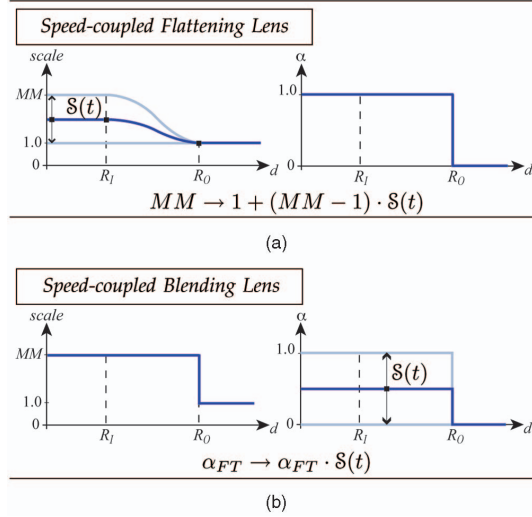


Fig. 7. Lenses with speed-coupled properties.

Other properties can be made speed dependent, including the radii R_I and R_O , as well as the transluence value in the lens' flattop α_{FT} . For instance, the SPEED-COUPLED BLENDING lens, which to our knowledge features the best focus targeting performance, is also easily obtained as shown in Fig. 7b. This lens features a larger flattop area compared to lenses of the same size that feature a transition zone. This makes the earlier mentioned focus targeting task easier for the user from a purely motor perspective, but the occlusion stemming from the absence of a smooth transition zone counterbalances this theoretical advantage. The occlusion problem is addressed by coupling α_{FT} to the speed of the lens: the lens becomes increasingly translucent as it is moved faster, and conversely (see Section 5 for more detail).

5 USER STUDY: FOCUS TARGETING PERFORMANCE

In order to verify the above assumptions about the properties of various lenses and their impact on user performance, we ran a series of experiments that evaluate lens usability in different navigation situations. We briefly present in this section the results of studies reported in detail in [13], that compare a set of lenses on an elementary action involved in any navigation task: *focus targeting*. The task consists in putting a given target in the flattop of the lens and is one of the building blocks of many higher level navigation tasks such as searching [35]. We then report in Section 6 the results of new experiments that further test the two most efficient lenses in a more complex navigation task that involves both *global* and *local* navigation actions.

5.1 Apparatus

In all our experiments, we used a Dell Precision 380 equipped with a 3 GHz Pentium D processor, an NVidia Quadro FX4500 graphics card, a $1,600 \times 1,200$ LCD monitor (21") and a Dell optical mouse. The program was written with the multiscale 2D framework presented in Section 7.1. The application was limited to a $1,400 \times 1,000$ window with

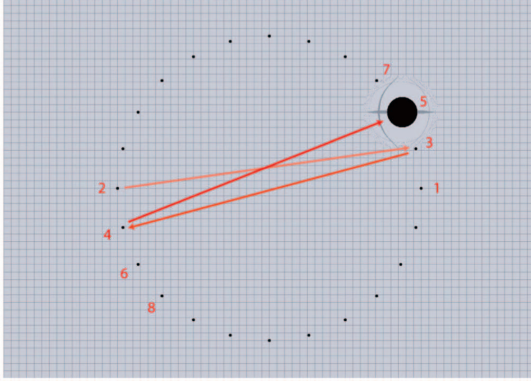


Fig. 8. Targets' order of appearance in a trial (targets are twice their actual relative size for legibility purposes).

a black padding of 100 pixels in order to accommodate instruction messages.

5.2 Task and Procedure

Our focus targeting task consisted in acquiring a target in the flattop of the lens as quickly as possible. In our experimental setting, the lens was centered on the mouse cursor. The task ended when the participant clicked the left mouse button, provided that the target was fully contained within the flattop. Each trial consisted in performing 24 successive focus targeting tasks in a row. As illustrated in Fig. 8, the targets were laid out in a circular manner. The order of appearance forced participants to perform focus targeting tasks in every direction, as recommended by the ISO9241-9 standard [36]. We decided to have only one target visible at a time, as we noticed during a pilot experiment in which all targets were visible that some participants were often taking advantage of the layout pattern to acquire the current target object by positioning the lens relative to that object's neighbors.

5.3 Experiment 1: Lens Type and Focus Targeting

We first compared the *focus targeting* performance and limits of five lenses described earlier: a plain MAGNIFYING GLASS, a simple distortion lens (FISH-EYE), BLENDING, SPEED-COUPLED FLATTENING, and SPEED-COUPLED BLENDING (see Section 4). Focus targeting performance was evaluated at five different magnification factors (MM). Higher magnification factors make the task increasingly difficult: 1) the transition area becomes harder to understand as it must integrate a larger part of the world in the same rendering area, and 2) it becomes harder to precisely position the target in the flattop of the lens, the latter being controlled in the motor space of the context window. To test the limits of each lens, we included factors up to $14\times$. Our experiment was a 5×5 within-participant design: each participant had to perform several trials using each of the five lenses with five different magnification factors ($MM \in \{2, 4, 6, 10, 14\}$). Ten volunteers (seven males, three females), from 23 to 40 years old (average 26.4, median 25), all with normal or corrected to normal vision, served in the experiment and allowed us to collect data on 11,500 actual focus targeting tasks.

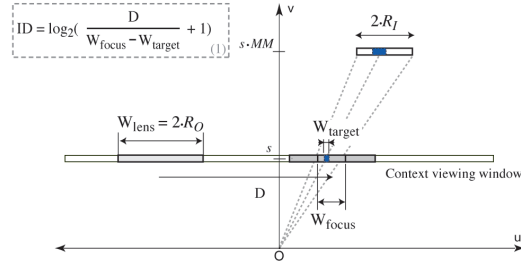


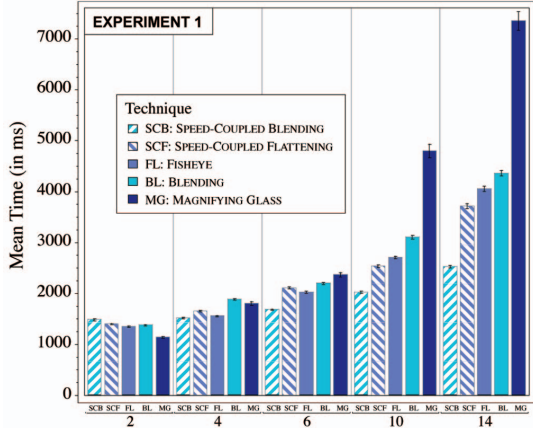
Fig. 9. Focus Targeting Task in a space scale diagram.

5.4 Results

Interestingly FISH-EYE and BLENDING do not significantly differ in their performance. We initially thought that translucence could improve user performance by eliminating the space-based transition drawbacks. Transitioning through space indeed introduces distortion that makes objects move away from the approaching lens focus before moving toward it very fast, making focus targeting difficult [33]. BLENDING does not actually overcome this problem since it introduces a new one: the high cognitive effort required to comprehend transitions based on gradually increasing translucence which, as opposed to distortion-based transitions, do not rely on a familiar physical metaphor.

We expected speed-based lenses (SPEED-COUPLED FLATTENING and SPEED-COUPLED BLENDING) to outperform their static versions (FISH-EYE and MAGNIFYING GLASS). Each focus targeting task can be divided into two phases: in the first phase, the user moves the lens quickly to reach the target's vicinity, while in the second phase, she moves it slowly to precisely position the target in the focus. In the first phase, the user is not interested in, and can actually be distracted by, information provided in the focus region since she is trying to reach a distant object in the context as quick as possible. By smoothly and automatically neutralizing the focus and transition regions during this phase, and then restoring them, speed-based lenses should help the user. Our results did actually support that this is the case for SPEED-COUPLED BLENDING and MAGNIFYING GLASS: smoothly neutralizing and restoring the focus of a MAGNIFYING GLASS by making it translucent does improve performance. However, our participants were not significantly faster with SPEED-COUPLED FLATTENING than with FISH-EYE. This was especially surprising since the study conducted in [33] showed a significant improvement in users performance with SPEED-COUPLED FLATTENING. We think this inconsistency is probably due to implementation differences: we implemented SPEED-COUPLED FLATTENING as a constrained lens while it was implemented as a full-screen lens by Gutwin. In full-screen lenses, distortion affects the whole representation, which thus benefits more from the neutralization effect than constrained lenses that only affect a limited area.

From a pure motor perspective, the difficulty of a focus targeting task can be evaluated as a view pointing task in a fixed-scale interface [37]. We can thus use (1) in Fig. 9 to quantify the difficulty of moving the lens' flattop, of size

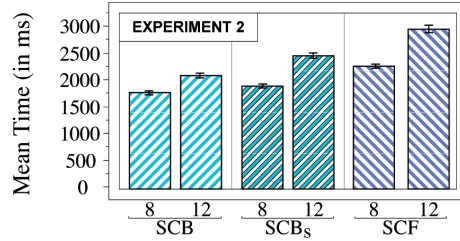
Fig. 10. Mean completion time per *Technique* $\times$ *MM* condition.

W_{focus} , to a position where it will contain the target, of size W_{target} , initially located at a distance D from the lens' center. Formula (1) computes the Index of Difficulty, ID , of our focus targeting task. The lens' position in the context window is controlled in the visual and motor space of that window. W_{target} , W_{focus} and D are thus expressed in context pixels: in our experiment, $W_{target} = 8$ pixels and $D = 800$ pixels, while W_{focus} depends on a given *Lens* $\times$ *MM* condition: $W_{focus} = (2 \times R_I)/MM$. As *MM* increases, the size of W_{focus} decreases, making the task more difficult. For lenses of equal size (R_O), the size of the flattop (R_I), and thus W_{focus} , vary depending on the lens type. MAGNIFYING GLASS and SPEED-COUPLED BLENDING lenses are made of a flattop only: $W_{focus} = W_{lens} = 200$, while other lenses have to accommodate the transition within the same overall area: $W_{focus} = W_{lens}/2 = 100$ in our implementation. MAGNIFYING GLASS and SPEED-COUPLED BLENDING thus feature a larger flattop than other lenses with the same overall size, consequently making focus targeting easier from a motor perspective: ID ranges from 3.2 to 6.3 for MAGNIFYING GLASS and SPEED-COUPLED BLENDING while it ranges from 4.2 to 8 for FISH-EYE, SPEED-COUPLED FLATTENING, and BLENDING. Our data showed that SPEED-COUPLED BLENDING is actually faster than all other lenses starting at $MM = 4$. However, MAGNIFYING GLASS becomes the worst lens at $MM = 6$: its large opaque flattop causes occlusion that makes the second phase of the task (precise positioning) too difficult to make users benefit from a larger flattop.

Fig. 10 summarizes these results. Our analyses only provide a partial order of performance between the five lenses but strongly support that SPEED-COUPLED BLENDING lens is the most efficient lens.

5.5 Experiment 2: Flat Top Size and Focus Targeting

Experiment 1 compared lenses with the same size (R_O). We found that SPEED-COUPLED BLENDING lenses outperform SPEED-COUPLED FLATTENING lenses, and attributed this performance gain 1) to the large flattop of the

Fig. 11. Mean completion time per *MM* $\times$ *Technique* condition.

SPEED-COUPLED BLENDING lens, which makes focus targeting easier from a motor perspective, and 2) to the absence of distortion and reduction of occlusion effects through the coupling of focus translucence with lens speed. Experiment 2 aimed at better understanding the results of the previous experiment by identifying the contribution of both properties to this performance gain. We studied how SPEED-COUPLED BLENDING performed at two "extreme" sizes: 1) the lens has the same size as other lenses ($W_{lens} = 200$), and 2) the lens has the same size as the flattop of lenses which accommodate a transition area and thus feature a smaller flattop ($W_{lens} = 100$), making focus targeting harder from a motor perspective as explained earlier. We called the latter SPEED-COUPLED BLENDING_{small} and compared it to SPEED-COUPLED BLENDING and SPEED-COUPLED FLATTENING, both from the previous experiment.

Two representative magnification factors were selected: $MM \in \{8, 12\}$. This experiment was thus a 3×2 within-participant design. Six volunteers (five males, one female), from 23 to 40 years old (average 27.8, median 25.5), all with normal or corrected to normal vision, served in the experiment. Fig. 11 illustrates our results: even at the same level of motor difficulty, the SPEED-COUPLED BLENDING lens still performs better than the SPEED-COUPLED FLATTENING lens. This means that interface designers are given several options to improve upon a classical lens such as FISH-EYE: 1) they can either get a smaller but more efficient lens (in terms of focus targeting performance), saving screen real estate for the context, 2) if the latter is not critical they can make the SPEED-COUPLED BLENDING lens occupy the same space as a FISH-EYE would, further improving focus targeting performance, or 3) find a balance between these solutions.

6 EXPERIMENT 3: TRANSLUCENCE AND SPEED-DEPENDENCE

Lieberman used translucence in Powers of Ten Thousands [38], a bifocal display technique that makes the focus and context views share the same physical screen space, by using multiple translucent layers. Even though it has been shown to be usable in exploratory studies [39], [40], this type of representation based on transparent or translucent layers is cognitively demanding, causing visual interferences that are the source of serious legibility problems, and requiring additional mental effort from the user to relate focus and

context. Translucence can hence affect targeting performance, especially when targets are superimposed on a complex background such as a map or photograph. Speed-dependent properties can also be confusing as they affect the lens' appearance depending on cursor movements. As the simple abstract world we used in the first two experiments might have hidden potential negative effects caused by translucence, we present here further controlled experiments aimed at verifying that our comparative lens performance ordering is still valid in more realistic environments.

6.1 Preliminary Experiment

In [13], we had already conducted a preliminary study to assess the potential effects of translucence on targeting performance. The task was the same as described earlier, but the 24 targets were laid out on a satellite photograph, and could either be filled with a fully opaque red color or with a translucent red, in which case they blended into the background and were less easily identifiable. The satellite photograph was a $7,000 \times 5,000$ pixels portion of NASA's Blue Marble world map [34], providing an appropriate level of detail in both the focus and context regions. This experiment yielded a performance ordering consistent with that observed in the first two. Target opacity had a significant effect on performance only for BLENDING. This result is not unexpected as the BLENDING lens can be prone to visual interference between focus and context in the transition region depending on the nature of the representation, especially when noncontrasted objects are targeted. No matter how aesthetically pleasing (several participants noted that it produced very nice graphical renderings), the BLENDING lens suffers from a lack of reliance on a familiar physical metaphor, and proneness to visual interference in the transition region. The SPEED-COUPLED BLENDING lens, however, does not seem to suffer from these problems, as its use of translucence is very different: it can be seen as a magnifying glass whose content smoothly fades out to prevent occlusion at focus targeting time.

In the remainder of this section, we further evaluate speed-dependent lenses by introducing a task that implies more speed variations when operating the lens by forcing both *global navigation* (large movement followed by a stop) and *local navigation* (small movements for fine-grain positioning), as the preliminary experiment described above suggests that SPEED-COUPLED BLENDING could cause some legibility problems since it is neither opaque nor fully transparent during local navigation phases.

6.2 Task and Procedure

Transitions based on distortion or transparency add complexity to the representation, and can affect usability differently depending on the nature of the objects displayed. To test lens usability in a wide range of realistic situations, we used different representations. We conducted two experiments illustrated in Fig. 12 based on two different types of representation: a network (vector graphics) for Experiment 4_{graph} ($Bg = graph$), and a high-resolution satellite map (bitmap) for Experiment 4_{map} ($Bg = map$). In both cases, the task is the same. A word is displayed to the user in three locations: top, center, and bottom of the screen. Participants are instructed to memorize this word as they will have to search for it in the representation. Once this

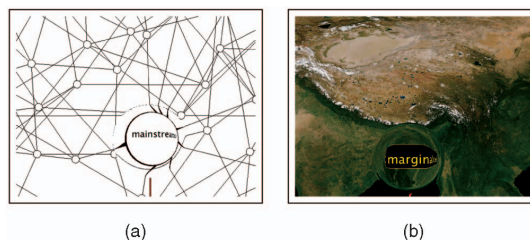


Fig. 12. (a) Experiment 4_{graph} (labels displayed in black) and (b) Experiment 4_{map} (labels displayed in yellow over a black background). Viewports are cropped to show details.

target word is memorized, participants put the cursor on a red square (20×20 pixels) located at the center of the screen and press the space bar to start the trial. The red square disappears, as does the target word displayed at the center (the ones displayed in the top and bottom margins remain displayed throughout the trial in case participants forget it). Words appear successively in the same locations as the circular targets did in previous experiments. When a word appears, the participant has to move the lens over it to be able to read it (a word represents about 27×8 pixels in the context and is thus unreadable at this scale). If the word is not the target word, she proceeds to the next one by pressing the space bar. This is repeated until she finds the target word, in which case she clicks the left mouse button to end the current trial and start a new one.

In both cases (mouse click or space bar), the word must be *in focus*. For a word to be considered in focus, our software uses the following criterion: the intersection area between the word and the lens' flattop should be at least 66 percent of the flattop area (thus ensuring that the word can be read). A word can, however, never be fully displayed in the flattop. This is to force participants to perform local navigation as described earlier. Here again to compare lenses both in usual and extreme conditions, we use two magnification factors ($MM \in \{8, 12\}$). Font size is set to 42 pts (at context scale) for $MM = 8$ and 28 pts for $MM = 12$, so that the lens' flattop can display at most six letters at full magnification. We use two word lengths to test the effect of the amount of local navigation on lens performance ($LabLength \in \{8, 12\}$). To decrease the probability of a participant recognizing a word only based on a few specific letters (which would reduce the amount of local navigation), we have chosen words that are similar to a certain degree: all eight-letter words start with "a" and end by "ed;" all 12-letter words start with "m" and end by "ed." We also had to make participants believe that the target word could appear at any time so they don't turn the task into a pure routine motor task. To that end, we introduced a secondary factor, *Rank*, that was used to control how many objects participants had to inspect before finding the target word for a given trial. *Rank* could take four different values: {2, 4, 6, 8}.

The graph of Experiment 4_{graph} contained 76 nodes that were laid out in a semirandom manner so as to provide a uniform density and to make sure that a node would coincide with each of the potential target locations (a word was always displayed inside a node, see Fig. 12a). For Experiment 4_{map} , we used the NASA world map mentioned

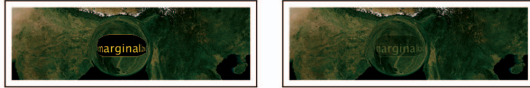


Fig. 13. Experiment 4_{map} : Opaque target versus Translucent target.

earlier (Fig. 12b). In this latter experiment, we introduced an additional factor: in half of the trials, words were opaque (O) while in the other half, they were translucent² (T) ($Opacity \in \{O, T\}$), as shown in Fig. 13. We hypothesized that background and focus might be perceptually interpreted as one illegible image if contrast is not strong enough, especially for SPEED-COUPLED BLENDING as it itself makes use of translucence. *Opacity* was not included as a factor in Experiment 4_{graph} because sharp edges displayed on a uniform background are strongly contrasted.

Twelve volunteers (nine males, three females), from 23 to 33 years (average 26.7, median 26.5), all with normal or corrected to normal vision, no color blindness, served in the experiment. Each of them was involved in both experiments. Experiment 4_{graph} lasted around 40 minutes, and 4_{map} lasted around 1 hour (instructions were shorter since all participants were already familiar with the task). The two experiments were performed on two different days to minimize fatigue and boredom. Each experiment was composed of four blocks, one per *Lens* $\times$ *MM* condition. Successive changes of *Lens* values would have been too disturbing. To counterbalance *Lens* presentation order, six participants saw the two SPEED-COUPLED FLATTENING blocks first, while the six other participants started with SPEED-COUPLED BLENDING. We always presented block *MM* = 8 first for a given *Lens*, so as to avoid harder conditions being presented first to participants.

A *Lens* $\times$ *MM* block contained four series of four trials in Experiment 4_{graph} and eight series of four trials in Experiment 4_{map} . A series contained words of the same length and presented the four different *Rank* values in a pseudorandom order to ensure that the overall difficulty was the same for all participants. In Experiment 4_{graph} , a participant saw alternatively a series of 8-letter words and a series of 12-letter words, twice. In Experiment 4_{map} , a participant saw alternatively two series of 8-letter words and two series of 12-letter words, twice; one series per opacity value (*Opacity* = O then *Opacity* = T). Table 1 summarizes this experimental design.

6.3 Results and Discussion

We collected three main measures for our analyses: 1) completion *Time*, i.e., the time interval between the appearance of the first word and the click on the target word; 2) the number of *Reading* errors, i.e., when the participant notices that she has pressed the space bar instead of clicking on the target word or if she has visited an abnormally large number of words³; 3) the number of

2. When *Opacity* = T , words and their black background are rendered with an alpha channel set to 0.2, thus blending into the satellite image, which makes them more difficult to identify (Fig. 13).

3. Instructors told participants that if they had visited about 15 words without having seen the target word this meant they had missed it. In this case, the participants could press the Escape key to skip the current trial and restart it (the timer was reset).

TABLE 1
Summary of Design for Experiment 3

Exp. 4_{graph}	Exp. 4_{map}
2 <i>Lens</i>	2 <i>Lens</i>
$\times$ 2 <i>MM</i>	$\times$ 2 <i>MM</i>
$\times$ 2 <i>LabLength</i>	$\times$ 2 <i>LabLength</i>
	$\times$ 2 <i>Opacity</i>
$\times$ 4 <i>Rank</i>	$\times$ 4 <i>Rank</i>
$\times$ 2 replications	$\times$ 2 replications
$\times$ 12 participants	$\times$ 12 participants
= 768 trials	= 1536 trials

Acquisition errors, i.e., when the participant presses the space bar or clicks while the word is *not in focus*. In this case, the message “target not in focus” would flash and the participant would have to adjust the lens (the trial continues, the timer is not reset).

A total of 62 trials among 768 were restarted because of *Reading* errors in Experiment 4_{graph} (~ 8 percent) and 128 trials among 1,536 in Experiment 4_{map} (~ 9 percent). Note that the recorded time for these particular trials was potentially biased, as participants could avoid some cognitive aspects of the task and turn it into a simple motor task. For instance, if a participant remembered the rank at which the target word appeared when realizing her failure, she could avoid having to carefully read the intermediary words the second time, simply pointing at them. To avoid analyzing data with an unbalanced number of measures per factor, we left these trials in our data after having checked that 1) *Reading* errors were uniformly distributed among the primary factors and 2) that neither primary factors nor *Rank* had a significant effect on the number of *Reading* errors.

Figs. 14a, 14b, 14c, 14d, 14e, 14f, 14g, and 14h show the data collected along the *Lens*, *MM*, and *LabLength* factors. Regarding *Time*, we did not observe any significant effect of any condition. We can see in the first two columns that participants were faster using SPEED-COUPLED BLENDING than SPEED-COUPLED FLATTENING. This difference was, however, not statistically significant.

Differences in accuracy were stronger. In both experiments, participants were more accurate using SPEED-COUPLED BLENDING than SPEED-COUPLED FLATTENING. *Lens* and *MM* had a significant effect on both *Time* and *Acquisition* errors (Table 2). Furthermore, differences between lenses in terms of accuracy increased with the magnification factor (Figs. 14c and 14d): *Lens* $\times$ *MM* had a significant effect on *Acquisition* errors.

Participants were more accurate with shorter words: *LabLength* had a significant effect on *Acquisition* errors in Experiment 4_{map} ($F_{1,11} = 22, p < 0.0001$), see Fig. 14h. However, participants were not significantly faster with shorter words. We did not expect this observation since the task is supposed to be harder from a motor perspective. The local navigation required by longer words probably penalizes overall task performance more than the motor aspects involved in a simple focus targeting task. In addition, lenses seem to be unequally affected by word length: *Lens* $\times$ *LabLength* was significant on *Acquisition* errors ($F_{1,11} = 17, p < 0.0001$) in Experiment 4_{map} . These results tend to show that SPEED-COUPLED BLENDING better supports local navigation than SPEED-COUPLED

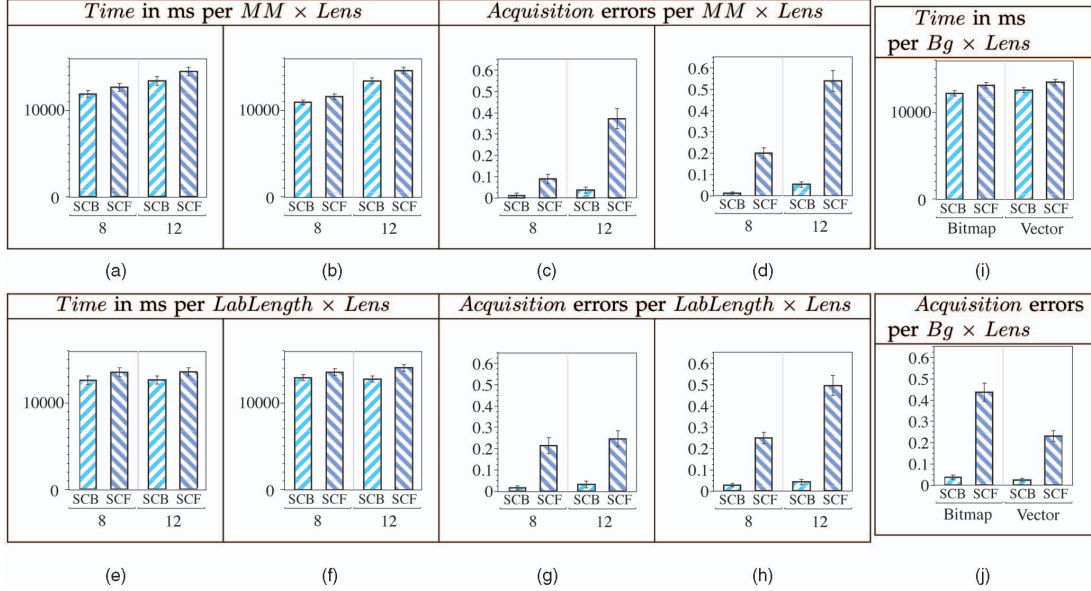


Fig. 14. (a), (c), (e), and (g) Experiment 4_{graph} and (b), (d), (f), and (h) Experiment 4_{map} : Time and Acquisition errors. SCB = Speed-Coupled Blending lens; SCF = Speed-Coupled Flattening lens.

FLATTENING (Figs. 14f and 14h). This latter effect, observed only in Experiment 4_{map} , reinforces our intuition that lens usability is affected by the type of representation. To have a closer look at this effect, we built a table resulting from the concatenation of trials of Experiment 4_{graph} ($Bg = Graph$) and trials where $Opacity = O$ of Experiment 4_{map} ($Bg = Map$). Analysis of variance revealed a significant effect of Bg on Acquisition errors ($F_{1,11} = 17, p < 0.0001$) but not on Time (Figs. 14i and 14j). Participants were more accurate on a vector-based representation (the network) probably because the white background helped perceive the limits of the label. We also found one interaction effect $Lens \times Bg$ on Acquisition errors ($F_{1,11} = 14, p = 0.0002$): surprisingly SPEED-COUPLED FLATTENING is more penalized by background type than SPEED-COUPLED BLENDING is (left of Fig. 14j). Transparency does not hinder performance on complex representations, and seems more robust than distortion for high magnification factors. Hence, our hypotheses about usability problems due to the use of transparency are not supported by this experiment. Analysis of the $Opacity$ factor in Experiment 4_{map} also supports SPEED-COUPLED BLENDING's robustness. The effect of $Opacity$ on Acquisition errors was significant ($F_{1,11} = 6, p = 0.01$), which is consistent with

participants remarks about the reading difficulty they had when labels were translucent. Interaction effect $Lens \times Opacity$ was also significant on Acquisition errors ($F_{1,11} = 5, p = 0.02$) revealing that users were more strongly affected by label opacity with SPEED-COUPLED FLATTENING than with SPEED-COUPLED BLENDING. As a summary, speed-coupled translucence does not have a negative impact on reading performance. The dynamically varying translucence of the SPEED-COUPLED BLENDING lens does not cause significant visual interference, even on complex scenes featuring a low level of contrast. The SPEED-COUPLED BLENDING lens remains more efficient than the SPEED-COUPLED FLATTENING lens.

7 IMPLEMENTATIONS

According to the previous evaluations, Sigma Lenses should prove useful in various types of graphical user interfaces. The next step consists in investigating how to add support for the framework in existing graphics environments. Though implementation is not straightforward, the approach is generic enough that it can be implemented in different environments with only a few requirements. The underlying graphics library must allow 1) for the scene to be rendered at different levels of detail, and 2) for the pixels that constitute the two rendered images (the context region and the lens region) to be manipulated and composited before the actual rendering to the screen occurs. The following sections describe two different implementations, one for multiscale 2D graphics, the other for OpenGL scenes.

7.1 Multiscale 2D Framework

Pad [1] was one of the first toolkits designed for the implementation of multiscale interfaces. Since then, several

TABLE 2
Significant Effects Revealed by Analysis of Variance of MM and $Lens$ Factors on Acquisition Errors

	Exp. 4_{graph}	Exp. 4_{map}
$Lens$	$F_{1,11} = 54, p < .0001$	$F_{1,11} = 150, p < .0001$
MM	$F_{1,11} = 32, p < .0001$	$F_{1,11} = 48, p < .0001$
$Lens \times MM$	$F_{1,11} = 20, p < .0001$	$F_{1,11} = 29, p < .0001$

other zoomable user interface toolkits have been developed, including Piccolo [2] and ZVTM [3]. Our implementation is based on the latter, an open source toolkit built on top of Java2D. It allows for the same scene to be rendered from different viewpoints. Furthermore, as the toolkit does not rely on Java's internal double buffering mechanism but implements its own and makes the offscreen rendering buffers publicly accessible, adding support for Sigma Lenses was easy. Our extension works directly on these offscreen images, and is independent of their content. It is thus readily compatible with all graphical objects that can be displayed by the toolkit, including arbitrary vector-based shapes, bitmap images, and text rendered with any font.

A call to the extension is simply inserted in the rendering loop, between the rendering of the main offscreen buffer, which corresponds to the context region, and its copy to the screen. The extension asks for a second rendering from the viewpoint of a camera set to observe the lens region. Clipping algorithms internal to the toolkit make sure that only objects visible through each camera get projected and drawn in the associated offscreen buffer. Our extension then creates the focus+context representation. As a significant part of the final rendering will match the context buffer exactly, the latter serves as the target of the transformation to prevent unnecessary copy operations. For each pixel (x, y) in the subregion corresponding to the lens, we simply read a pixel in the raster associated with the lens buffer using an index computed through function $\mathcal{R}$, thus achieving magnification as well as distortion depending on the drop-off defined by $\mathcal{G}_{scale}$. This pixel gets composited with the original pixel (x, y) from the context buffer, using alpha blending if necessary with a value of α computed through function $\mathcal{R}$ (more specifically $\mathcal{G}_{comp}$). Pixels outside the lens region are left untouched.

The core classes of the extension represent about 700 lines of actual code, supporting three different `BufferedImage` types (type varies depending on operating system and color depth). Taking advantage of the inheritance mechanism, a new lens is then typically written with less than 100 lines of code: most of those lines are dedicated to constructors, getters, and setters. Lines that actually modify the rendering of the transition, i.e., implementations of $\mathcal{G}_{scale}$ and $\mathcal{G}_{comp}$ specific to each lens, only account for about 10 percent of those 100 lines except for lenses with speed-dependent properties, in which case the coupling with the interpolated low-pass filter adds extra 80 lines of code.

Performance. We measured the performance of this implementation on a representation consisting of a multi-scale version of the high-resolution $(86,400 \times 43,200)$ pixels Blue Marble Next Generation world map from NASA [34]. The pyramid consists of 2,728 tiles, each $1,350 \times 1,350$ pixels in size. We overlaid shapefile data representing country boundaries as vector graphics on top of the map, amounting to 23,715 segments forming 1,375 polygons of various shapes and sizes. Performance tests were run on a Windows XP PC, equipped with a 3 GHz Pentium D processor, 2 GB of RAM and an NVIDIA Quadro FX 4500 graphics card with 512 MB of memory, using Java 1.6 with the default DirectX rendering pipeline enabled. The application ran in a $1,280 \times 640$ window. All lenses were 200 px wide, occupying about

5 percent of the rendering area, and were tested at various magnification factors ranging from $2\times$ to $8\times$. Rendering the content of the context buffer took an average of 23 ms (base condition). Average time spent rendering and compositing the content of the lens buffer ranged from 24 ms in the best case ($2\times$ distortion lens) to 74 ms in the worst case ($8\times$ hovering lens), corresponding to overall frame rates varying between 21 and 10 fps. While not extremely high, the measured frame rates show that our framework can be implemented in graphics environments that do not benefit from significant hardware acceleration such as plain Java2D, and still achieve interactive frame rates for relatively complex representations.

7.2 3D Framework

The second implementation takes advantage of programmable graphics hardware. Lenses are written with the OpenGL Shading Language (GLSL). The rendering process is as follows: First, the lens region is rendered to a texture, thanks to a frame buffer object that makes it possible to render to other destinations than those provided by the window system. The projection frustum is set to match the lens region, so that only objects visible in that region get rendered in this first pass. Only the projection frustum is changed, not the camera position, so as not to introduce any discontinuity between the context and lens regions. The viewport's dimensions, and thus those of the target texture are set⁴ according to the lens buffer's size: $2 \cdot MM \cdot R_O \times 2 \cdot MM \cdot R_O$. Second, the viewport and perspective projections are set according to the dimensions of the application's window to render the scene as it would look like in the absence of any lens. Third, the texture generated during the first step is mapped on a plane matching the lens region using normalized texture coordinates. If the scene were to be rendered through the standard fixed function pipeline, this higher resolution texture would perfectly blend into the context. Instead, we use a fragment shader that implements the lens' displacement and compositing functions. The data contained in the texture are accessed through this fragment shader. Context fragment color data are accessed from the lens texture through a texture sampler by using the texel coordinates calculated by the standard fixed function pipeline in previous stages of the rendering process:

```
//focus color data access with
texture2D(lensTex, vec2(x*1.0/texWidth,
y*1.0/texHeight));

//context color data access with
texture2D(lensTex, gl_TexCoord[textureNum].st);
```

The fragment shader is independent of the content of the scene observed. It is an implementation of the specific instance of function $\mathcal{R}$ defining one particular lens, nothing more. External data controlled by the application, such as the position of the lens (x_c, y_c) , its magnification factor MM , or the cursor speed, are sent to the shader using GLSL

4. If the dimensions exceed the maximum texture size permitted by the graphics card, the lens region can be rendered in smaller tiles that are then read using the multitexture access capabilities of fragment shaders.

uniform variables. The source code of a stand-alone shader is thus very small, typically 60-80 lines, instructions related to a specific instance of $\mathcal{G}_{scale}$ or $\mathcal{G}_{comp}$ typically amounting to 2-4 lines.

Performance. We used the same hardware configuration as in Section 7.1, except that the software was running under Linux, with version 169.07 of the NVIDIA driver. The application ran in a $1,050 \times 750$ window and displayed 3D objects of varying complexity. We tested lenses 200-800 px wide, at magnification factors ranging from $2\times$ to $8\times$. Measures indicate that, in these ranges, the values of MM and R_O have negligible influence on performance. In all cases, rendering the content of the lens region takes approximately the same amount of time as rendering the context in the frame buffer. For instance, a 3D model made of 69,451 facets is displayed at 123 fps with a lens versus 231 fps with no lens; another model made of 345,944 facets is displayed at 28 fps versus 53 fps. Performance could be improved if necessary by implementing manual frustum culling, taking the different sizes of the context and lens regions into account (lens rendering time would depend on the lens region's dimensions). Time spent distorting and compositing in the fragment shader is not significant compared to scene rendering time.

8 DISCUSSION AND FUTURE WORK

As opposed to most techniques described in the literature, we wanted to develop a lightweight, representation-independent approach to the problem of focus+context navigation based on constrained lenses. This choice has implications in terms of implementation effort, expressive power and graphics performance.

Graphics performance will obviously vary depending on the complexity of the representation and on the underlying graphics framework's capabilities, such as clipping algorithms and level of support for hardware acceleration; but even in cases where most of the computations are done on the software side, acceptable frame rates can be achieved for most lenses on reasonably complex scenes such as the one described in Section 7.1. Performance can, however, be an issue when highly magnifying lenses defined by complex drop-off functions for both scale and translucence are used in a non-hardware-accelerated framework. In that case, values for $\mathcal{G}_{scale}$ (displacement function) and $\mathcal{G}_{comp}$ (blending gradient) can be precomputed for each target pixel in the lens region and stored in a data structure such as an array or texture for fast lookup. Memory consumption will amount to a maximum of two structures storing $(2R_O)^2$ floating point numbers, and can be lowered to a quarter of that value by taking advantage of vertical and horizontal symmetry for lenses based on $L(P)$ -metrics.

The shape of a lens does not necessarily have to feature the above-mentioned symmetry. While most constrained lenses have been restricted to regular shapes such as circles and squares (Magic Lenses [41] excepted), irregular perimeters can be obtained fairly easily by defining them using parametric equations that make R_I and R_O angle dependent. We are planning to better integrate such perimeters in the framework, our final goal being to build *adaptive lenses* whose shape changes to provide more

relevant magnifications of the objects in focus by accessing information about their geometry.

One strong point of our approach is its good expressive power which comes at a relatively small implementation cost. The two implementations described in Section 7 show that it can be implemented with minimal effort in various graphics frameworks, only requiring offscreen drawing capabilities, a feature commonly available in graphics APIs, and the possibility to draw the representation at two different scales. Once the core lens programming elements are in place (when required at all), writing a new lens typically takes only a few lines, most of the code being the same from one lens to another, and differences residing mainly in the definition of transitions.

Working on the rendered scene, distorting and compositing the focus and context regions by manipulating pixels makes our technique fully independent of the actual 2D or 3D objects that constitute the representation. However, this approach also has some limitations. One limitation is that performance depends on the magnification factor and size of the lens, as a consequence of the increased level of detail in the focus. Another issue is related to the quality of the rendering in the transition area for lenses that make use of distortion: as points get compressed in the transition region, small objects such as points may be lost, and lines cut if they fall between pixels, especially for high magnification factors. This problem can be partially addressed by specifying an expression of $\mathcal{G}_{comp}$ that matches $\mathcal{G}_{scale}$, but this, in turn, can introduce unwanted rendering artifacts. Solutions based on MIP maps [42] can help address this issue, but will be computationally expensive if the content of the representation in the magnified region varies continuously.

9 SUMMARY

We have presented an approach to the implementation of constrained magnification lenses for navigation in large workspaces, based on the displacement and compositing of pixels from two renderings at different scales. This technique has several practical advantages such as its independence with respect to the type of representation and its low cost of implementation. But, more importantly, it allows for a variety of dynamic transitions between the focus and context regions, including—but not limited to—those covered in the Sigma Lens framework. This framework enables the design of new lenses, such as the SPEED-COUPLED BLENDING lens, which has been shown to perform very efficiently through a series of controlled experiments that cover cognitive aspects of navigation tasks ranging from motor performance to legibility issues on various types of representations.

REFERENCES

- [1] K. Perlin and D. Fox, "Pad: An Alternative Approach to the Computer Interface," *Proc. ACM SIGGRAPH '93*, pp. 57-64, 1993.
- [2] B.B. Bederson, J. Grosjean, and J. Meyer, "Toolkit Design for Interactive Structured Graphics," *IEEE Trans. Software Eng.*, vol. 30, no. 8, pp. 535-546, Aug. 2004.
- [3] E. Pietriga, "A Toolkit for Addressing HCI Issues in Visual Language Environments," *Proc. IEEE Symp. Visual Languages and Human-Centric Computing (VL/HCC '05)*, pp. 145-152, 2005.
- [4] C. Ware and S. Osborne, "Exploration and Virtual Camera Control in Virtual Three Dimensional Environments," *Proc. Symp. Interactive 3D Graphics (SI3D '90)*, pp. 175-183, 1990.

- [5] W.C. Donelson, "Spatial Management of Information," *Proc. ACM SIGGRAPH '78*, pp. 203-209, 1978.
- [6] J. Lamping, R. Rao, and P. Pirolli, "A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies," *Proc. SIGCHI Conf. Human Factors in Computing Systems (CHI '95)*, pp. 401-408, 1995.
- [7] T. Munzner, F. Guimbretière, S. Tasiran, L. Zhang, and Y. Zhou, "Treejuxtaposer: Scalable Tree Comparison Using Focus+Context with Guaranteed Visibility," *Proc. ACM SIGGRAPH '03*, pp. 453-462, 2003.
- [8] E.R. Gansner, Y. Koren, and S.C. North, "Topological Fisheye Views for Visualizing Large Graphs," *IEEE Trans. Visualization and Computer Graphics*, vol. 11, no. 4, pp. 457-468, July/Aug. 2005.
- [9] M.S.T. Carpendale, J. Ligh, and E. Pattison, "Achieving Higher Magnification in Context," *Proc. ACM Symp. User Interface Software and Technology (UIST '04)*, pp. 71-80, 2004.
- [10] G. Shoemaker and C. Gutwin, "Supporting Multi-Point Interaction in Visual Workspaces," *Proc. SIGCHI Conf. Human Factors in Computing Systems*, pp. 999-1008, 2007.
- [11] G. Ramos, A. Cockburn, R. Balakrishnan, and M. Beaudouin-Lafon, "Pointing Lenses: Facilitating Stylus Input through Visual- and Motor-Space Magnification," *Proc. SIGCHI Conf. Human Factors in Computing Systems (CHI '07)*, pp. 757-766, 2007.
- [12] P. Blenkhorn, G. Evans, A. King, S.H. Kurniawan, and A. Sutcliffe, "Screen Magnifiers: Evolution and Evaluation," *IEEE Computer Graphics and Applications*, vol. 23, no. 5, pp. 54-61, Sept./Oct. 2003.
- [13] E. Pietriga and C. Appert, "Sigma Lenses: Focus-Context Transitions Combining Space, Time and Translucence," *Proc. 26th CHI Conf. Human Factors in Computing Systems (CHI '08)*, pp. 1343-1352, 2008.
- [14] G.G. Robertson and J.D. Mackinlay, "The Document Lens," *Proc. ACM Symp. User Interface Software and Technology (UIST '93)*, pp. 101-108, 1993.
- [15] M. Sarkar, S.S. Snibbe, O.J. Tversky, and S.P. Reiss, "Stretching the Rubber Sheet: A Metaphor for Viewing Large Layouts on Small Screens," *Proc. ACM Symp. User Interface Software and Technology (UIST '93)*, pp. 81-91, 1993.
- [16] M.S.T. Carpendale, D.J. Cowperthwaite, and F.D. Fracchia, "3-Dimensional Pliable Surfaces: For the Effective Presentation of Visual Information," *Proc. ACM Symp. User Interface Software and Technology (UIST '95)*, pp. 217-226, 1995.
- [17] T.A. Keahey and E.L. Robertson, "Nonlinear Magnification Fields," *Proc. 1997 IEEE Symp. Information Visualization (INFOVIS '97)*, pp. 51-58, 1997.
- [18] M.S.T. Carpendale and C. Montagnese, "A Framework for Unifying Presentation Space," *Proc. ACM Symp. User Interface Software and Technology (UIST '01)*, pp. 61-70, 2001.
- [19] C. Gutwin and A. Skopik, "Fisheyes are Good for Large Steering Tasks," *Proc. SIGCHI Conf. Human Factors in Computing Systems (CHI '03)*, pp. 201-208, 2003.
- [20] A. Cockburn, A. Karlson, and B.B. Bederson, "A Review of Overview+Detail, Zooming, and Focus+Context Interfaces," *ACM Computing Surveys*, vol. 41, no. 1, pp. 1-31, 2008.
- [21] E. LaMar, B. Hamann, and K.I. Joy, "A Magnification Lens for Interactive Volume Visualization," *Proc. Ninth Pacific Conf. Computer Graphics and Applications (PG '01)*, 2001.
- [22] Y. Yang, J.X. Chen, and M. Beheshti, "Nonlinear Perspective Projections and Magic Lenses: 3D View Deformation," *IEEE Computer Graphics and Applications*, vol. 25, no. 1, pp. 76-84, Jan./Feb. 2005.
- [23] J. Brosz, F.F. Samavati, M.T.C. Sheelagh, and M.C. Sousa, "Single Camera Flexible Projection," *Proc. Fifth Int'l Symp. Non-Photorealistic Animation and Rendering (NPAR '07)*, pp. 33-42, 2007.
- [24] A. Angelidis and K. Singh, "Space Deformations and Their Application to Shape Modeling," *Proc. ACM SIGGRAPH '06 Courses*, pp. 10-29, 2006.
- [25] A.H. Barr, "Global and Local Deformations of Solid Primitives," *Proc. ACM SIGGRAPH '84*, vol. 18, no. 3, pp. 21-30, 1984.
- [26] T.W. Sederberg and S.R. Parry, "Free-Form Deformation of Solid Geometric Models," *Proc. ACM SIGGRAPH '86*, vol. 20, no. 4, pp. 151-160, 1986.
- [27] P. Rademacher, "View-Dependent Geometry," *Proc. ACM SIGGRAPH '99*, pp. 439-446, 1999.
- [28] C.D. Correa and D. Silver, "Programmable Shaders for Deformation Rendering," *Proc. 22nd ACM SIGGRAPH/EUROGRAPHICS Symp. Graphics Hardware (GH '07)*, pp. 89-96, 2007.
- [29] S. Schein, E. Karpen, and G. Elber, "Real-Time Geometric Deformation Displacement Maps Using Programmable Hardware," *Visual Computer*, vol. 21, no. 8, pp. 791-800, 2005.
- [30] Y. Kurzion and R. Yagel, "Interactive Space Deformation with Hardware-Assisted Rendering," *IEEE Computer Graphics and Applications*, vol. 17, no. 5, pp. 66-77, Sept. 1997.
- [31] M. Spindler, M. Bubke, T. Germer, and T. Strothotte, "Camera Textures," *Proc. Fourth Int'l Conf. Computer Graphics and Interactive Techniques in Australasia and Southeast Asia (GRAPHITE '06)*, pp. 295-302, 2006.
- [32] T. Porter and T. Duff, "Compositing Digital Images," *Proc. ACM SIGGRAPH '84*, pp. 253-259, 1984.
- [33] C. Gutwin, "Improving Focus in Interactive Fisheye Views," *Proc. SIGCHI Conf. Human Factors in Computing Systems (CHI '02)*, pp. 267-274, 2002.
- [34] R. Stockli, E. Vermote, N. Saleous, R. Simmon, and D. Herring, "The Blue Marble Next Generation—A True Color Earth Dataset Including Seasonal Dynamics from MODIS," NASA Earth Observatory, 2005.
- [35] E. Pietriga, C. Appert, and M. Beaudouin-Lafon, "Pointing and Beyond: An Operationalization and Preliminary Evaluation of Multi-Scale Searching," *Proc. Human Factors in Computing Systems (CHI '07)*, pp. 1215-1224, 2007.
- [36] ISO, "9241-9 Ergonomic Requirements for Office Work with Visual Display Terminals (VDTs)-Part 9: Requirements for Non-Keyboard Input Devices," Int'l Organization for Standardization, 2000.
- [37] Y. Guiard and M. Beaudouin-Lafon, "Target Acquisition in Multiscale Electronic Worlds," *Int'l J. Human-Computer Studies*, vol. 61, no. 6, pp. 875-905, Dec. 2004.
- [38] H. Lieberman, "Powers of Ten Thousand: Navigating in Large Information Spaces," *Proc. ACM Symp. User Interface Software and Technology (UIST '94)*, pp. 15-16, 1994.
- [39] D.A. Cox, J.S. Chugh, C. Gutwin, and S. Greenberg, "The Usability of Transparent Overview Layers," *Proc. Conf. Summary on Human Factors in Computing Systems (CHI '98)*, pp. 301-302, 1998.
- [40] B.L. Harrison, H. Ishii, K.J. Vicente, and W.A.S. Buxton, "Transparent Layered User Interfaces: An Evaluation of a Display Design to Enhance Focused and Divided Attention," *Proc. Conf. Human Factors in Computing Systems (CHI '95)*, pp. 317-324, 1995.
- [41] E.A. Bier, M.C. Stone, K. Pier, W. Buxton, and T.D. DeRose, "Toolglass and Magic Lenses: The See-Through Interface," *Proc. ACM SIGGRAPH '93*, pp. 73-80, 1993.
- [42] L. Williams, "Pyramidal Parametrics," *ACM SIGGRAPH Computer Graphics*, vol. 17, no. 3, pp. 1-11, 1983.

Emmanuel Pietriga received the PhD degree in computer science from the Institut National Polytechnique de Grenoble (France) in 2002. He worked for INRIA and Xerox Research Centre Europe, did his postdoctoral research at the Massachusetts Institute of Technology (MIT) as a team member of the World Wide Web Consortium, and is now back at INRIA as a research scientist. He works on interaction techniques and tools for interactive structured graphics environments such as multiscale user interfaces.

Olivier Bau received the MS degree in computer science from the Université Joseph Fourier (Grenoble, France) in 2006. He is currently working toward the PhD degree in the Department of Computer Science at Université Paris-Sud (France). His research interests include human-computer interaction, computer graphics, and design.

Caroline Appert received the PhD degree in computer science from the Université Paris-Sud (France) in 2007 and did her postdoctoral research at IBM Almaden (California). She is now a permanent research scientist at CNRS (France). Her past and current work focuses on the design of tools for programming and evaluating advanced interaction techniques.

► For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/publications/dlib.

CHI 2010: Interfaces and Visualization

April 10–15, 2010, Atlanta, GA, USA

High-Precision Magnification Lenses

Caroline Appert^{1,2}
appert@lri.fr

Olivier Chapuis^{1,2}
chapis@lri.fr

Emmanuel Pietriga^{2,1}
emmanuel.pietriga@inria.fr

¹LRI - Univ. Paris-Sud & CNRS
Orsay, France

²INRIA
Orsay, France

ABSTRACT

Focus+context interfaces provide in-place magnification of a region of the display, smoothly integrating the focus of attention into its surroundings. Two representations of the data exist simultaneously at two different scales, providing an alternative to classical pan & zoom for navigating multi-scale interfaces. For many practical applications however, the magnification range of focus+context techniques is too limited. This paper addresses this limitation by exploring the *quantization* problem: the mismatch between visual and motor precision in the magnified region. We introduce three new interaction techniques that solve this problem by integrating fast navigation and high-precision interaction in the magnified region. *Speed* couples precision to navigation speed. *Key* and *Ring* use a discrete switch between precision levels, the former using a keyboard modifier, the latter by decoupling the cursor from the lens' center. We report on three experiments showing that our techniques make interacting with lenses easier while increasing the range of practical magnification factors, and that performance can be further improved by integrating speed-dependent visual behaviors.

Author Keywords

Focus+Context, Lenses, Quantization, Navigation, Selection

ACM Classification Keywords

H. Information Systems H.5 Information Interfaces and Presentation H.5.2 User Interfaces (H.1.2, I.3.6)

General Terms

Design, Human Factors

INTRODUCTION

Although display technologies continue to increase in size and resolution, datasets are increasing even faster. Scientific data, e.g., telescope images and microscope views of the brain, and generated data, e.g., network visualizations, geographical information systems and digital libraries, are too big to be displayed in their entirety, even on very large wall-sized displays. In Google Maps, the ratio between extreme scales is about 250,000. Vast gigapixel images, such as the 400,000-pixel wide image of the inner-part of our galaxy from the Spitzer telescope also require huge scale

factors between a full overview and the most detailed zoom. Users do not necessarily need to navigate through the entire scale range at one given time, but still, they need interaction techniques that will allow them to fluidly navigate between focused and contextual views of large datasets. Such techniques are typically based on the following interface schemes [8]: overview + detail, zooming, focus + context; none of which offers an ideal solution. The task determines which technique is most appropriate, taking scale range, the nature of the representation, input device, available screen real-estate, and of course, the user's preferences, into account.

This paper introduces techniques designed to improve lens-based focus+context interfaces. Our goals are to extend the range of practical magnification factors, which is currently very limited, and to make low-level interactions easier. For the sake of clarity, we illustrate all of our techniques with one common type of lens: constrained magnification lenses [4, 18, 19]. However, our improvements are generic and apply to all types of lenses. They can also be adapted to other focus+context interfaces, including hyperbolic trees [16] and stretchable rubber sheets [20].

QUANTIZATION IN FOCUS+CONTEXT INTERFACES

Constrained lenses provide in-place magnification of a bounded region of the representation (Figure 1-a). The *focus* is integrated in the *context*, leaving a significant part of the latter unchanged. Typical examples of such lenses include magnifying glasses and many distortion-oriented techniques

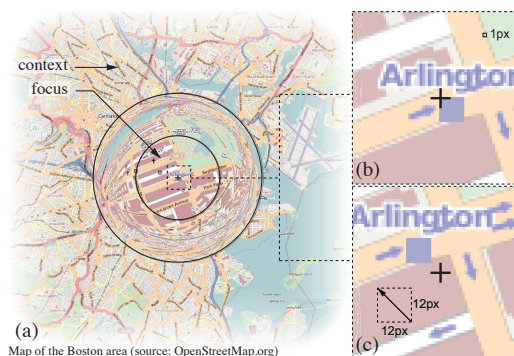


Figure 1. (a) In-place magnification by a factor of 12; (b) center of magnified region with cursor in the middle (detail); (c) same region after moving the lens by one pixel both South and East.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CHI 2010, April 10 – 15, 2010, Atlanta, Georgia, USA

Copyright 2010 ACM 978-1-60558-929-9/10/04...\$10.00.

CHI 2010: Interfaces and Visualization

April 10–15, 2010, Atlanta, GA, USA

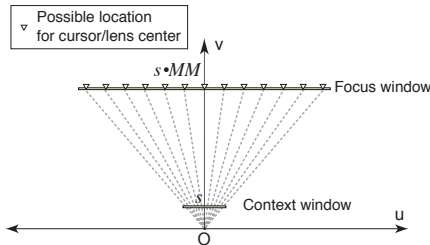


Figure 2. Space-scale diagram of possible locations for lens center (each ray corresponds to one pixel in context space).

such as the so-called graphical fisheyes. Early implementations of magnification techniques only magnified the pixels of the context by duplicating them without adding more detail, thus severely limiting the range of useful magnification factors (up to 4x). Newer implementations [4, 18] do provide more detail as magnification increases. Theoretically, this means that any magnification factor can be applied, if relevant data is available. In practice, this is not the case as another problem arises that gets worse as magnification increases: *quantization*.

Lenses are most often coupled with the cursor and centered on it. The cursor, and thus the lens, are operated at context scale. This allows for fast repositioning of the lens in the information space, since moving the input device by one unit makes the lens move by one pixel at context scale. However, this also means that when moving the input device by one unit (dot), the representation in the magnified region is offset by MM pixels, where MM is the focus' magnification factor. This means that only one pixel every MM pixels can fall below the cursor in the magnified region. In other words some pixels are unreachable, as visual space has been enlarged in the focus region but motor space has not.

This problem is illustrated in Figure 1: between (b) and (c), the lens has moved by 1 unit of the input device, corresponding to 1 pixel in the context, but the magnified region is offset by 12 pixels. Objects can thus be difficult or even impossible to select; even if their visual size is above what is usually considered a small target (less than 5 pixels). The square representing Arlington station in Figure 1 is 9-pixel wide, yet its motor size is only 1 pixel.

Figure 2 illustrates the problem with a space-scale diagram [11]: the center of the lens can only be located on a pixel in the focus window that is *aligned* – on the same ray in the space-scale diagram – with a pixel in the context window. Since the focus window is MM^2 larger than the context window, and since the cursor is located at the lens' center, only one out of MM^2 pixels can be selected. Figure 2 shows that as MM increases, more pixels become unreachable.

Beyond the general problem of pixel-precise selection in the magnified region, quantization also hinders focus targeting, i.e., the action that consists in positioning the lens on the object of interest [12, 18]. This action gets harder as the magnification factor increases, even becoming impossible at extreme magnification factors.

This quantization problem has limited the range of magnification factors that can be used in practice; the upper limit reported in the literature rarely exceeds 8x, a value relatively low compared to the ranges of scale encountered in the information spaces mentioned earlier.

In this paper, we introduce techniques that make it possible to perform both fast navigation for focus targeting and high-precision selection in the focus region in a seamless manner, enabling higher magnification factors than those allowed by conventional techniques. After an overview of related work, we introduce our techniques. *Speed* continuously adapts motor precision to navigation speed. *Key* and *Ring* use a discrete switch between two levels of precision (focus and context), the former using an additional input channel, the latter by decoupling the cursor from the lens' center. We then report the results of two controlled experiments that evaluate focus targeting and object selection performance. Finally, we iterate our designs by integrating speed-dependent visual behaviors from the Sigma Lens framework [18]. The resulting hybrid lenses further improve performance, as shown in a third controlled experiment.

RELATED WORK

Most techniques for navigating multi-scale information spaces are based on either overview + detail, zooming or focus + context (see Cockburn et al. [8] for a very thorough survey). Zooming interfaces, e.g., [21, 14] display a single level of scale and therefore require a temporal separation to transition between “focus” and “context” views. They usually do not suffer from quantization effects, but both views cannot be observed simultaneously. Overview+detail interfaces [13, 22] show both views simultaneously using spatial separation, still requiring some mental effort to integrate the two views. They usually allow pixel-precise selections in the detail region, but focus targeting is also subject to quantization problems in conventional bird's eye views.

Focus+context techniques “aim to decrease the short term memory load associated with assimilating distinct views of a system” [8] by integrating the focus region inside the context. This integration, however, limits the range of magnification factors of practical use. Basic magnifying glasses occlude the surroundings of the magnified region [12]. To address this issue, distortion oriented techniques provide a smooth transition between the focus and context views. Distortion, however, causes problems for focus targeting and understanding of the visual scene. Carpendale et al. [4] describe elaborate transitions that enhance the rendering of the distorted area and make higher magnifications comprehensible from a visual perspective. Gutwin's *Speed-coupled flattening lens* [12] cancels distortion when the lens is repositioned by the user, thus removing a major hindrance to focus targeting. The Sigma Lens framework [18] generalizes the idea of speed-coupling to a larger set of lens parameters. For example, the *Speed-coupled blending lens* makes focus targeting easier from a motor perspective by increasing the focus region's size for the same overall lens size, using a dynamically varying translucence level to smoothly transition between focus and context.

CHI 2010: Interfaces and Visualization

Although their primary goal is different, focus+context interfaces share issues with techniques designed to facilitate pointing on the desktop. The decoupling of visual and motor spaces plays a central role in techniques designed to facilitate the selection of small targets, e.g., [6, 7, 17] – see [2] for a detailed survey. Not designed for exploratory multi-scale navigation, but closer to our problem are pointing lenses [19], which punctually enlarge both visual and motor space to facilitate small target selection through stylus input. However, visual space is enlarged by duplicating the pixels of the original representation. The popup vernier [1] enables users to make precise, sub-pixel adjustments to the position of objects by transitioning from coarse to fine-grain dragging mode through an explicit mode switch. The technique provides visual feedback based on the metaphor of vernier calipers to make precise adjustments between both scales.

LENSES WITH HIGH-PRECISION MOTOR CONTROL

The quantization effect is due to the mismatch between visual and motor space precision in the focus region. This mismatch, in turn, is caused by the following two properties of conventional lenses:

- (P1) the cursor is located at the center of the lens, and
- (P2) the cursor location is controlled in context space.

These properties cause problems with the two low-level actions performed by users: *focus targeting*, and *object selection* within the magnified region. In this section we introduce three techniques that address these problems by breaking the above properties.

For all our techniques, lens displacements of less than MM focus pixels, corresponding to displacements of less than 1 context pixel, are achieved by slightly moving the representation in the focus region while keeping the cursor stationary (see discussion of Experiment 2's results for more detail).

Precision through Mode Switching: the *Key* technique

The first approach to address the problem is to provide a way of controlling the location of the lens in focus space (as opposed to context space). We immediately discard the solution that consists in solely interacting in focus space because of obvious performance issues to navigate moderate to large distances (all distances are multiplied by MM in focus space). The simplest technique uses two control modes: a *context speed* mode and a *focus speed* mode. This requires an additional input channel to perform the mode switch, for instance using a modifier key such as SHIFT. Users can then navigate large distances at *context speed*, where one input device unit is mapped to one context pixel, i.e., MM focus pixels, and perform precise adjustments at *focus speed*, where one input device unit corresponds to one focus pixel.

Figure 3 illustrates this technique, called *Key*: the first case (No modifier) is represented by the topmost grey line; the second case (Shift pressed) by the bottommost grey line. When SHIFT is pressed, (P2) is broken. A similar “precision mode” is already available in, e.g., Microsoft Office to freely position objects away from the intersections formed by the underlying virtual grid using a modifier key.

April 10–15, 2010, Atlanta, GA, USA

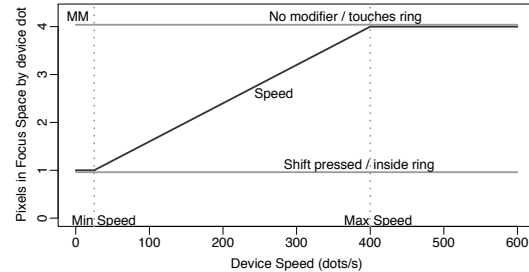


Figure 3. Displacement in focus space (in pixels) for one input device unit move in function of the input device speed ($MM = 4$).

The *Key* technique represents a simple solution. However, as the selection tools based on Magic Lenses [3], an additional channel is required to make the explicit mode switch. Bi-manual input techniques are still uncommon. Modifier keys tend to be used for other purposes by applications, and their use often results in a “*slightly less than seamless interaction style*” [2]. The next two techniques we propose do not require any additional input channel.

Speed-dependent Motor Precision: the *Speed* technique

Following recent works that successfully used speed-dependent properties to facilitate pointing [5] and multi-scale navigation [12, 14, 18], our first idea was to map the precision of the lens control to the input device's speed with a *continuous* function, relying on the assumption that a high speed is used to navigate large distances while a low speed is more characteristic of a precise adjustment (as observed for classical pointing [2]).

The black line (*Speed*) in Figure 3 illustrates the behavior of our speed-dependent precision lens. Cursor instant speed s is computed as the mean speed over the last four move events. It is mapped to the lens' speed so as to break (P2) as follows:

- (i) if $s < MIN_SPEED$ then the lens moves at *focus speed* ;
- (ii) if $MIN_SPEED \leq s \leq MAX_SPEED$ then the lens moves by x focus-pixels for 1 input device unit, where x is $1 + (1 - \frac{MAX_SPEED - s}{MAX_SPEED - MIN_SPEED}) \times (MM - 1)$;
- (iii) if $s > MAX_SPEED$ then the lens moves at *context speed* like a conventional lens.

Cursor-in-flat-top Motor Precision: the *Ring* technique

The last technique is inspired by Tracking menus [10]. Consider a large rigid ring (e.g., a bracelet) on a flat surface (e.g., a desk). The ring can be moved by putting a finger inside it and then moving that finger while keeping it in contact with the surface to pull the ring. This is the basic metaphor used to interact with the *Ring* lens: the ring is the lens' focus region (called the *flat-top*) and the cursor is the finger.

The *Ring* lens breaks property (P1): it decouples the cursor from the lens center; the cursor can freely move *within* the flat-top at *focus scale*, thus enabling pixel-precise pointing in the magnified region (bottommost grey line (*Inside ring*) in Figure 3). When the cursor comes into contact with the flat-top's border, it pulls the lens at *context speed*, enabling fast repositioning of the lens in the information space (topmost

CHI 2010: Interfaces and Visualization

grey line (Pushing ring) in Figure 3). Figure 5 illustrates the lens behavior when the cursor comes into contact with the ring: the segment joining the lens center (g) to the contact point (p) is progressively aligned with the cursor's direction.

Decoupling the cursor's location from the lens' center has a drawback when changing direction: because the user has to move the cursor to the other end of the flat-top before she can pull the lens in the opposite direction. We tried to address this issue by pushing the physical metaphor: we introduced friction in the model to make the ring slide when the cursor stops, with the effect of repositioning the lens' center so as to match the cursor's position. We were not able however to get a satisfying result, and abandoned the idea.

EXPERIMENTS

We conducted two experiments to compare the performance and limits of the three lenses described above. Participants were asked to perform a simple task: selecting an object in the magnified area. The targets were laid out in a circular manner and the order of appearance forced participants to perform the task in every direction, following the recommendations of the ISO 9241-9 standard [9]. Only one target was visible at a time so that participants could not take advantage of the layout to facilitate the task: as soon as the participant clicked on one target, the next target appeared. The recorded movement time is the interval between the appearance of the target and a click on it. The target is presented as a yellow circle on a gray background, and is always surrounded by a 10-pixel red square clearly visible in the context view. The background is also decorated by a grid to help participants understand the transition between context and focus view, and to minimize desert fog effects [15] that can occur with scenes that are too uniform.

Analysis of the Task

A *pointing task* with a lens is typically divided in two main phases: (i) *focus targeting*, which consists in putting a given target inside the flat-top of the lens (Figure 4-(a) and (b)) and (ii) *cursor pointing* to precisely position the cursor over the target (Figure 4-(b) and (c)).

The *focus targeting* task has an index of difficulty of about:

$$ID_{FT} = \log_2 \left(1 + \frac{D_c}{(W_{FT_c} - W_c)} \right)$$

where W_{FT_c} and W_c are the respective sizes of the flat-top and the target in context pixels, and D_c is the distance to the target in context pixels as well¹. This formula clearly shows that difficulty increases as distance increases, as the size of the flat-top decreases, and as the size of the target decreases. The size of the flat-top in context pixels is directly related to the magnification factor of the lens, MM . Indeed, the size of the flat-top is fixed in terms of focus pixels, so the higher MM , the smaller the size of the magnified area in context pixels (see [18] for an analysis of the difficulty of a focus targeting task).

¹ ID_{FT} is the exact index of difficulty when the target must be fully contained in the flat-top. Here the task is slightly easier because the target just has to intersect the flat-top.

April 10–15, 2010, Atlanta, GA, USA

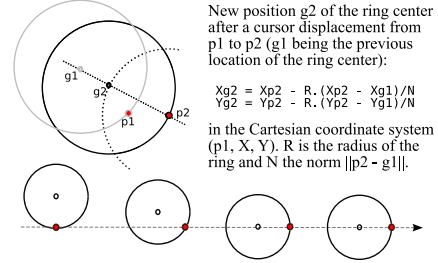


Figure 5. Bottom: behavior of a *Ring* lens when the cursor comes into contact with the flat-top's border at the bottom of the ring and then moves to the right. Top: Computation of the ring's location.

The final *cursor pointing* task mainly depends on the area of the target in focus space that intersects the flat-top after the focus targeting task. The larger this area, the easier the cursor pointing task. We can at least consider the best case, i.e., when the target is fully contained in the flat-top. In this case, the difficulty of the cursor pointing task can be assessed by the ratio $\frac{D_f}{W_f}$ where D_f is the distance between the cursor and the target, and W_f is the motor size of the target when magnified in the flat-top. The distance D_f is small, i.e., smaller than the flat-top's diameter, so we assume that the difficulty of the cursor pointing task is mainly caused by the value of W_f . Note that for regular lenses, the value of W_f is actually the size of the target at context scale because the target is only visually magnified. With our lenses however, since pixel-precise selections are possible, W_f is the magnified size of the target (at focus scale). We provide additional details about the division between the two subtasks in the following sections.

The first experiment tests pointing tasks with an average level of difficulty, while the second one tests pointing tasks with a very high level of difficulty, involving targets smaller-than-a-pixel wide at context scale. Our experimental design involves the three factors that determine the pointing task difficulty introduced above: the distance to the target (D_c), its width (W_c), and the lens' magnification factor MM .

Experiments: Apparatus

We conducted the experiments on a desktop computer running Java 1.5 using the open-source ZVTM toolkit. The display was a 21" LCD monitor with a resolution of 1600 x 1200 (≈ 100 dpi). The mouse was a regular optical desktop mouse at 400 dpi with the default acceleration function.

Experiment 1: Design

The goal of the first experiment is to test whether any of the three techniques we introduced in the previous section degrade performance when compared with regular lenses (*Reg*). We expect them to improve overall performance because the overall task difficulty is theoretically lower. On the one hand, the focus targeting task should not be harder: since we test small targets with lenses having the same flat-top size, the distance in context space is the main factor contributing to difficulty. All our lenses are able to navigate large distances like a regular lens, i.e., move at *context speed* (*Key*:

CHI 2010: Interfaces and Visualization

April 10–15, 2010, Atlanta, GA, USA

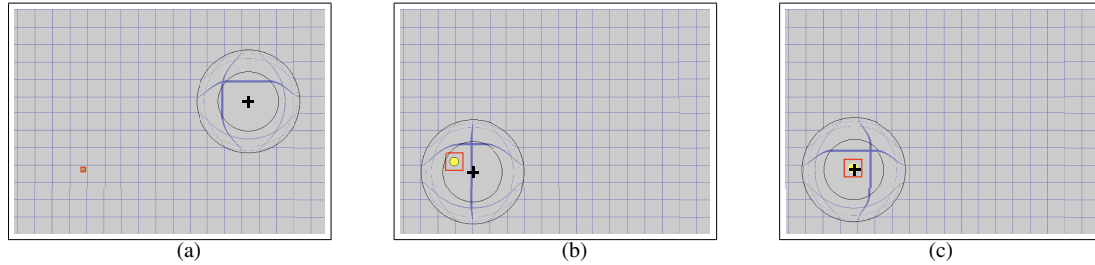


Figure 4. Screenshots of our experimental task: *focus targeting* from (a) to (b) and, *cursor pointing* from (b) to (c). Screenshots have been cropped to show details, and cursors have been made thicker to improve readability.

when SHIFT is released; *Ring*: when the cursor pulls the lens; *Speed*: when the lens moves fast enough). On the other hand, cursor pointing should be easier since the difficulty of this second phase mainly depends on the target's motor width in focus space. Since all of our lenses allow to navigate at *focus speed*, they can take benefit of the magnified target size whereas this is not the case with a regular lens: even though it is magnified, the target size in motor space is the same as if it were not magnified.

Sixteen unpaid volunteers (14 male, 2 female), age 20 to 35 year-old (average 26.8, median 26), all with normal or corrected to normal vision, served in Experiment 1. Experiment 1 was a $4 \times 2 \times 2 \times 3$ within-subject design with the following factors:

- Technique: $TECH \in \{Speed, Key, Ring, Reg\}$
- Magnification: $MM \in \{4, 8\}$
- Distance between targets (context pixels): $DC \in \{400, 800\}$
- Target width (context pixels): $WC \in \{1, 3, 5\}$

We grouped trials into four blocks, one per technique ($TECH$), so as not to disturb participants with too many changes between lenses. The presentation order was counterbalanced across participants using a Latin square. Within a $TECH$ block, each participant saw two sub-blocks, one per value of magnification factor (MM). The presentation order of the two values of MM was also counterbalanced across techniques and participants. For each $TECH \times MM$ condition, participants experienced a series of 12 trials per $DC \times WC$ condition, i.e., 12 targets laid out in a circular pattern as described earlier. We used a random order to present these $2 \times 3 = 6$ series within a sub-block. We removed the first trial of each series from our analyses as the cursor location is not controlled when a series begins. To summarize, we collected $4 \times 2 \times 2 \times 3 = 48$ series of 12 trials per $TECH \times MM \times DC \times WC$ condition, i.e., 16 participants $\times$ 48 series = 8448 trials for analysis. Before each $TECH$ condition, the experimenter took 2-3 minutes to explain the technique to be used next. Participants were told each time the value of MM was about to change, and had to complete 4 series of practice trials for each new $TECH \times MM$ condition.

Experiment 1: Results and Discussion

Our analysis is based on the full factorial model:

$$TECH \times MM \times WC \times DC \times \text{Random}(\text{PARTICIPANT})$$

with the following measures:

- *FTT*, the focus targeting time;

- *CPT*, the cursor pointing time;
- $MT = FTT + CPT$, the time interval between the appearance of the target and a successful mouse press on it (this measure includes penalties caused by errors); and
- *ER*, the error rate (an error is a press outside the target).

Analysis of variance reveals an effect of $TECH$ on MT ($F_{3,45} = 15.2, p < 0.0001$). A Tukey post-hoc test shows that *Reg* is the significantly slowest technique and that *Key* is significantly faster than *Ring*. Note that there is no significant difference between *Ring* and *Speed*, nor between *Speed* and *Key*. Participants also made more errors with *Reg* than with our techniques. We expected *Reg* to perform worse since, as we already mentioned, the target's motor size is in context pixels for *Reg* whereas it is in focus pixels for *Key*, *Speed* and *Ring*. The target is thus much harder to acquire in the *CPT* phase. Analysis of variance shows a significant effect of $TECH$ ($F_{3,45} = 18.5, p < 0.0001$) on *ER*. Figures 6-(a) and (b) respectively show the time MT and error rate *ER* for each $TECH \times WC$ condition.

We find a significant effect of DC ($F_{1,15} = 121.9, p < 0.0001$) on movement time MT . It is consistent with our expectations: DC has a significant effect on FTT ($F_{1,15} = 165, p < 0.0001$) while it does not on *CPT* ($p=0.4$). The higher the value of DC , the harder the focus targeting phase. Our techniques do not seem to be at a disadvantage in this phase compared to *Reg* since the effect of $DC \times TECH$ on FTT is not significant ($p=0.9$).

MM also has a significant effect on MT ($F_{1,15} = 249.6, p < 0.0001$), the effect being distributed across both FTT ($F_{1,15} = 515, p < 0.0001$) and *CPT* ($F_{1,15} = 79, p < 0.0001$). Figure 6-(c) shows the three measures per $TECH \times MM$: a bar represents MT per condition while the line shows the repartition between FTT (lower part of the bar) and *CPT* (upper part)². This clearly shows that a high MM leads to high FTT since the flat-top size in context pixels directly depends on MM , as explained in the previous section. A higher MM also means a larger target width in focus pixels. This can explain the effect of MM on *CPT*: *CPT* decreases as MM increases.

The target width in focus pixels is of course also related to WC , which is consistent with our observations: WC has an effect on both (i) FTT ($F_{2,30} = 45, p < 0.0001$) and (ii) *CPT* ($F_{2,30} = 1110, p < 0.0001$), and also on MT ($F_{2,30} =$

²Error bars in the figures represent the 95% confidence limits of the sample mean ($mean \pm StdErr \times 1.96$).

CHI 2010: Interfaces and Visualization

April 10–15, 2010, Atlanta, GA, USA

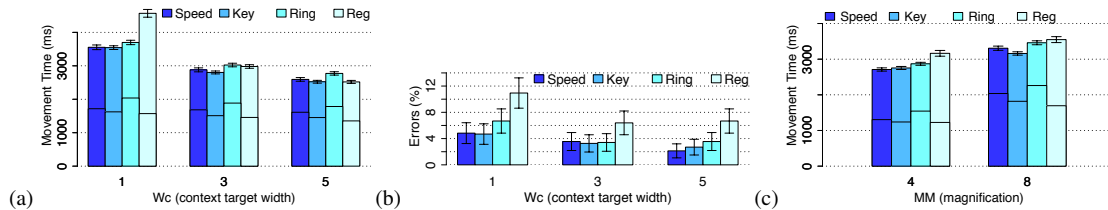


Figure 6. Movement time (a) and error rate (b) per TECH $\times$ WC. (c) Movement time per TECH $\times$ MM. For (a) and (c), the lower part of each bar represents focus targeting time, the upper part cursor pointing time.

623.8, $p < 0.0001$, Figure 6-(a)). Indeed, as we expected, the smaller Wc, the higher the focus targeting time (i). Also, the larger Wc, the larger the target in focus pixels to improve focus pointing time (ii). Regarding error rate, Wc ($F_{2,30} = 17.5$, $p < 0.0001$) and MM ($F_{1,15} = 16.8$, $p = 0.0009$) have a significant effect on ER: participants made more errors when the target size was small. This is a simple interpretation that explains the difference in means that we observe; but we have to refine it to reflect the more complex phenomenon that actually takes place.

Coming back to the effect of TECH, we also observe two significant interaction effects that involve TECH on MT.

First interaction effect: TECH $\times$ MM ($F_{3,45} = 4.7$, $p = 0.0063$) which can be observed on Figure 6-(c). A Tukey post-hoc test shows that for MM = 4, Speed, Key and Ring are significantly faster than Reg but this test also shows that for MM = 8, only Key and Speed are significantly faster than Reg (Ring no longer is). A closer look at the focus targeting phase explains why Ring seems to suffer from high magnification factors. We know that FTT increases as MM increases. We can observe on Figures 6-(c) and (a) that Ring is actually slower than the other techniques for this FTT phase. This is probably due to the cost of repairing overshoot errors during this phase: changes in direction are costly with Ring since the user first has to move the cursor to the opposite side of the flat-top before being able to pull the lens in the opposite direction.

Second interaction effect: TECH $\times$ WC ($F_{6,90} = 55.1$, $p < 0.0001$) which can be observed on Figure 6-(a). A Tukey post-hoc test shows a significant difference in mean for Wc=1 between Reg and the other techniques, while this difference is not significant for Wc=3 and Wc=5. To better assess the interpretation of such a result, we consider finer analyses on CPT. Figure 7 shows CPT for each TECH $\times$ MM $\times$ WC condition. Analyses reveal significant effects of TECH, MM and Wc and significant interactions TECH $\times$ MM and TECH $\times$ WC (all $p < 0.0001$) on CPT. Tukey post-hoc tests show that Key, Speed and Ring are globally faster than Reg for cursor pointing. This is not surprising since the motor size of the target is smaller for Reg than for the others, as we said earlier. However, this significant difference holds only for Wc=1 and Wc=3, not for Wc=5. In the latter case, only Speed is significantly faster than Reg. Moreover Ring is faster than Key for Wc=1, while Speed is not. These results suggest that Ring is particularly efficient for very small targets and that Speed is more appropriate for larger ones.

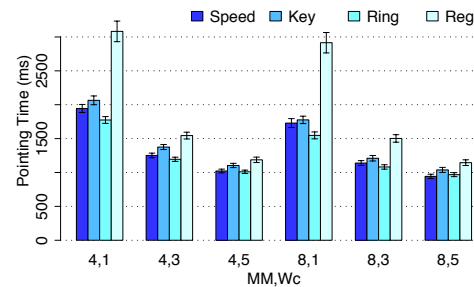


Figure 7. Cursor pointing time per TECH $\times$ MM $\times$ WC condition.

The latter observations suggest that modeling the movement time MT as the sum of FTT and CPT ($MT = FTT + CPT$) may be too naive to explain the subtle differences between techniques. For instance, this model does not explain the differences between Ring and Speed that depend on Wc. In the same spirit, we observe that the difference between Reg and other lenses for Wc=5 is very small considering that the target's motor size is 5 for Reg and 20 (MM=4) or 40 (MM=8) for Key, Speed and Ring. The additive model based also fails to explain the following observation: Speed features significantly higher FTT values than Key and Reg for MM=8 only. We tentatively explain this by the increased difficulty of controlling a lens with speed-dependent precision when the slope of the mapping function is too steep (linear function from MIN_SPEED to MAX_SPEED, i.e., focus speed to context speed on Figure 3). We tried several variations that, e.g., depend on the difference between these two speeds, without success. Using a gentler slope is frustrating because of the stickiness caused by the large movements required to reach the MAX_SPEED threshold. The more subtle differences we reported in the second part of this section may be explained by the fact that a transition phase between the focus targeting phase (FTT) and the cursor pointing phase (CPT) actually exists for our lenses: pressing a key for Key, stop pulling the flat-top for Ring, performing speed adjustments with Speed.

At the end of the experiment, participants were asked to rank the lenses (with ex-aequo allowed) using two criteria: perceived usability and performance. These two rankings were almost the same for all participants. All but one ranked Reg as their least preferred technique (one participant ranked it third with Speed fourth). There was no significant difference among other lenses. For instance, 8 participants ranked

CHI 2010: Interfaces and Visualization

Speed first, 3 ranked it second; 6 participants ranked *Key* first, 5 ranked it second, and 5 participants ranked *Ring* first, 7 ranked it second. We also asked participants to comment on the techniques. The main reason for the bad ranking of *Reg* is the great difficulty to acquire small targets, related to the *cursor jumping* effect due to quantization. Regarding *Speed*, most participants found the technique “*natural*”; some found the speed “*difficult to control*”. The participants who ranked *Key* high justified it by a “*transparent control*”; other participants complained about the need to use two hands. Regarding *Ring*, the cursor pointing phase was found easier because the lens is stationary, but participants also raised the overshooting problem discussed earlier.

To summarize, in comparison with regular lenses, precision lenses increase pointing accuracy. They also increase selection speed for small targets and are as fast for larger ones.

Experiment 2: Design

This second experiment evaluates our techniques on extreme tasks: very small target sizes and high magnification factors. We discard the *Reg* technique as it is not capable of achieving sub-pixel pointing tasks, i.e., involving targets that are smaller-than-a-pixel wide in context space. Another difference with Experiment 1 is that we use W_F as a factor instead of W_C . This allows us to isolate the effects of W_F and MM. Indeed, since $W_F = W_C \times MM$, two values of MM correspond to two different values of W_F for the same W_C value.

Twelve participants from Experiment 1 (10 male, 2 female), age 20 to 35 year-old (average 27.25, median 26.5), also served in Experiment 2. Experiment 2 was a $3 \times 2 \times 2 \times 3$ within-subject design with the following factors:

- $TECH \in \{Speed, Key, Ring\}$
- $MM \in \{8, 12\}$
- $DC \in \{400, 800\}$
- $W_F \in \{3, 5, 7\}$

As in Experiment 1, trials were blocked by technique, with presentation order counterbalanced across participants using a Latin square. The experimenter explained the technique to be used during 2-3 minutes before each $TECH$ condition. For each $TECH$, participants saw the two values of MM, grouped into two sub-blocks (sub-block presentation order were counterbalanced across techniques and participants). Each sub-block contained 6 series of 8 trials, 1 series per $DC \times MM$ condition, presented in a random order. To summarize, we collected $3 TECH \times 2 MM \times 2 DC \times 3 W_C \times (8-1) replications \times 12 participants = 3024$ trials for analysis. As in Experiment 1, participants were alerted by a message each time the MM value changed and had to complete 4 practice series for each $TECH \times MM$ condition.

Experiment 2: Results and Discussion

Our analysis is based on the full factorial model:

$$TECH \times MM \times W_F \times DC \times Random(PARTICIPANT)$$

We consider the same measures as in Experiment 1: task completion time MT , focus targeting time FTT , cursor pointing time CPT and error rate ER .

April 10–15, 2010, Atlanta, GA, USA

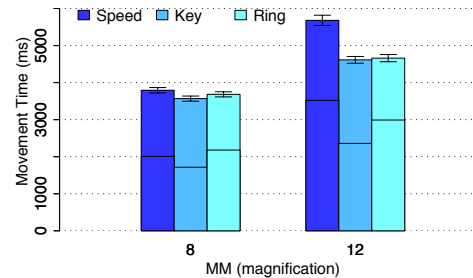


Figure 8. Movement time per $TECH \times MM$. The lower part of each bar represents focus targeting time, the upper part cursor pointing time.

Analysis of variance reveals simple effects of W_F ($F_{2,22} = 68$), MM ($F_{1,11} = 393$) and DC ($F_{1,11} = 65$) on MT (all $p < 0.0001$). As expected, MT increases as W_F decreases, as MM increases and as DC increases. Participants also make significantly more errors when W_F decreases (3.67% for $W_F = 7$, 5.36% for $W_F = 5$ and 8.82% for $W_F = 3$).

The differences in movement time MT among techniques is significant ($F_{2,22} = 21.6, p < 0.0001$) while the difference in error rate is not (6.15% for *Speed*, 6.05% for *Key* and 5.65% for *Ring*).

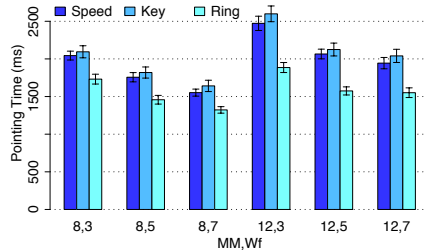
There is an interaction effect $TECH \times MM$ on MT ($F_{2,22} = 24.8, p < 0.0001$): Tukey post-hoc tests show that *Ring* and *Key* are significantly faster than *Speed* but only for $MM=12$ while these differences are not significant for $MM=8$. Figure 8 shows that this large difference at $MM=12$ is due to a sharp increase of focus targeting time (FTT) for *Speed*. Comments from participants confirm that the speed dependent control of motor precision is too hard when the difference between context scale and focus scale is too high, resulting in abrupt transitions. With *Speed*, participants did not succeed in controlling their speed: either they overshoot the target (targeting speed too high) or spent a lot of time putting the target in focus (speed too low). Therefore, *Speed* does not seem to be a suitable lens for pointing with a very high magnification factor: at $MM=12$, the linear function linking focus speed to context speed is too steep to be usable.

Figure 8 shows that focus targeting performance of *Ring* degrades as MM increases. However, good cursor pointing performance compensates for it, resulting in good overall task completion time. Figure 9 shows CPT for each $TECH \times MM \times W_C$ condition. Analysis of variance reveals a significant effect of W_F ($F_{2,22} = 230, p < 0.0001$) on CPT . As mentioned earlier, the larger W_F , the easier the cursor pointing task. However, the effects of MM ($F_{1,11} = 154, p < 0.0001$) and $TECH$ ($F_{2,22} = 64, p < 0.0001$) on CPT are less straightforward to interpret. CPT is higher when $MM=12$ than when $MM=8$, *Ring* is faster than *Key* and *Speed*, and the difference between *Ring* and both *Key* and *Speed* is larger when $MM=12$ than when $MM=8$ (the $TECH \times MM$ interaction is indeed significant on CPT , $F_{2,22} = 9.8, p = 0.0009$).

A plausible explanation for these effects lies in the differences in terms of Control-Display (C-D) gain among tech-

CHI 2010: Interfaces and Visualization

April 10–15, 2010, Atlanta, GA, USA

Figure 9. Cursor pointing time per TECH $\times$ MM $\times$ WF condition.

niques in the cursor pointing phase³. Figure 10 illustrates the difference in terms of control-display gain among lenses, all in high-precision mode. During the cursor pointing phase, *Ring* is stationary; only the cursor moves inside a static flat-top. This is not the case for *Key* and *Speed* for which high-precision cursor pointing is achieved through a combination of cursor movement and flat-top offset. In Figure 10, to achieve a mouse displacement of 15 units, the cursor has moved by 1 context pixel (= 8 focus pixels) and the representation has moved by 7 focus pixels to achieve an overall displacement of 15 focus pixels. As a result, the control-display gain is divided by MM for *Key* and *Speed*. This might be the cause for the observed performance degradation. This interpretation is consistent with the stronger degradation for *Key* and *Speed* than for *Ring* from MM=8 to MM=12. Note, however, that there is still a small degradation of CPT from MM=8 to MM=12 for *Ring*, that we tentatively explain by a harder focus targeting phase when MM=12 that influences the transition from focus targeting to cursor pointing.

To summarize, when pushed to extreme conditions, the *Speed* lens becomes significantly slower than the other precision lenses while *Ring* remains as fast as *Key* without requiring an additional input channel for mode switching.

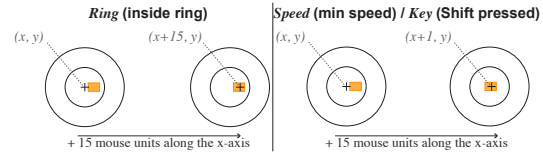
MOTOR CONTROL COMBINED WITH VISUAL FEEDBACK

Previous experiments show that techniques with advanced *motor* behaviors enable higher-precision focus targeting and object selection while increasing the upper limit of usable magnification factors. The Sigma Lens framework [18] takes a different approach at solving the same general problem by proposing advanced *visual* behaviors. We now explore how to combine these two orthogonal approaches to create hybrid lenses that further improve performance.

Sigma Lenses with High-Precision Motor Control

The two Sigma lens visual designs reported as the most efficient ones in [18] can be directly combined with our motor designs. The first one is the *Speed-coupled blending* (abbreviated *Blend*): it behaves as a simple magnifying glass whose transluence varies depending on lens speed. Smooth transition between focus and context is achieved through dynamic alpha blending instead of distortion. This enables a larger flat-top for the same overall lens size, reducing the

³The ratio between the distances traveled by the *cursor* and the *input device*, both expressed in metric units.

Figure 10. Difference in control-display gain between *Ring* and *Speed/Key* lenses (MM=8). In *italic*: cursor location on screen.

focus targeting task's index of difficulty. The other design (abbreviated *Flat*) is a variation on Gutwin's original *Speed-coupled flattening* [12]. The lens flattens itself into the context as its speed increases so as to eliminate the problems caused by distortion. Figure 11 illustrates both behaviors.

We designed four new techniques that result from the combination of one of the above two visual behaviors with either speed-dependent motor precision (*Speed*) or cursor-in-flat-top motor precision (*Ring*). *Key* was discarded because it proved awkward to combine explicit mode switching with speed-dependent visual properties.

Speed + Flat: this lens behaves like the original *Speed* design, except that the magnification factor decreases toward 1 as speed increases (Figure 11-a). The main advantage is that distortion no longer hinders focus targeting. Additionally, flattening provides indirect visual feedback about the lens' precision in motor space: it operates in context space when flattened, in focus space when not flattened.

Ring + Flat: This lens behaves like the original *Ring* design, with the magnification factor varying as above. As a consequence, the flat-top shrinks to a much smaller size (time stamp *t3* on Figure 11-a), thus making course corrections during focus targeting easier since the cursor is still restricted to that area. As above, distortion is canceled during focus targeting.

Ring + Blend: This distortion-free lens behaves like the original *Ring* design, except that the restricted area in which the cursor can evolve (the flat-top) is larger (time stamps *t1* and *t5* in Figure 11-b). As speed increases, the flat-top fades out, thus revealing the context during the focus targeting phase (time stamps *t2* to *t4*). An inner circle fades in, representing the region that will actually be magnified in the flat-top if the lens stops moving. The cursor is restricted to that smaller area, making course corrections less costly.

Speed + Blend: This lens behaves like the original *Speed* design without any distortion. As above, the flat-top fades out as speed increases and fades back in as speed decreases. Again, the larger flat-top reduces the focus targeting task's index of difficulty. In a way similar to *Speed + Flat*, blending provides indirect visual feedback about the lens' precision in motor space: it operates in context space when transparent, in focus space when opaque.

Experiment 3: Design

Our goal is to evaluate the potential benefits of combining techniques that enable higher motor precision with visual behaviors based on speed-coupling. We use *Static* versions,

CHI 2010: Interfaces and Visualization

April 10–15, 2010, Atlanta, GA, USA

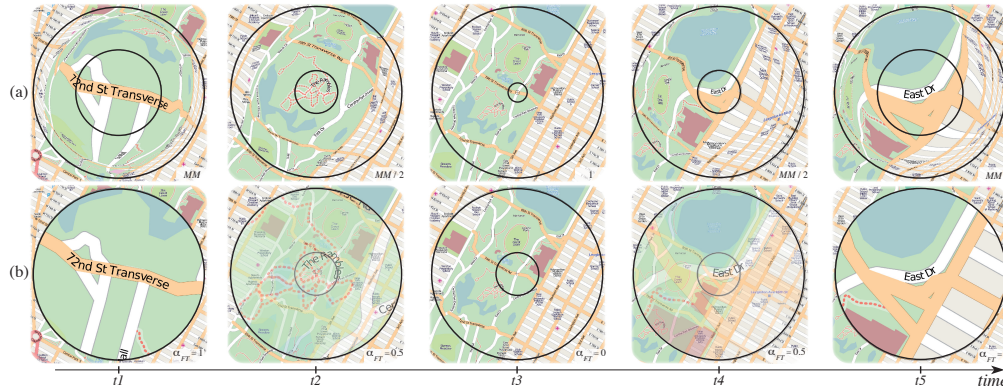


Figure 11. Behavior of two Sigma lenses during a focus targeting task ending on *East Drive* in Central Park. (a) As speed increases, the *speed-coupled flattening lens* smoothly flattens itself into the context (from $t1$ to $t3$), and gradually reverts to its original magnification factor when the target has been reached ($t4$ and $t5$). The inner circle delimits the region magnified in the flat-top. (b) As speed increases, the *speed-coupled blending lens* smoothly fades into the context (from $t1$ to $t3$), and gradually fades back in when the target has been reached ($t4$ and $t5$). The inner circle fades in as the lens fades out; it delimits which region of the context gets magnified in the lens. The magnification factor remains constant.

i.e., without any dynamic visual behavior, of our *Ring* and *Speed* techniques as baselines. Experiment 2 revealed that problems arise for the *difficult* tasks. We thus consider here difficult conditions in terms of magnification and target size. To reduce the length of the experiment, we discarded the *Dc* factor (distance between targets) as it did not raise any particular issue for any of the techniques.

Twelve participants from the previous experiments served in Experiment 3. Experiment 3 was a $2 \times 3 \times 2 \times 3$ within-subject design with the following factors:

- Motor precision technique: $TECH \in \{Speed, Ring\}$
- Visual behavior: $VB \in \{Blend, Flat, Static\}$
- Magnification: $MM \in \{8, 12\}$
- Target width in focus pixels: $WF \in \{3, 7, 15\}$

Trials were grouped into two main blocks, one per technique ($TECH$). These blocks were divided into three secondary blocks, one per visual behavior. The presentation order of $TECH$ main blocks and VB secondary blocks was counterbalanced across participants using a Latin square. Within a $TECH \times VB$ block, each participant saw two sub-blocks, one per magnification factor (MM); presentation order was counterbalanced as well. For each $TECH \times VB \times MM$ condition, participants experienced 3 series of 8 trials, one per value of WF , presented in a random order. We collected $2 \times 3 \times 3 \times 8 \times 12$ replications = 3024 trials for analysis. As with the other two experiments, participants received a short explanation before each $TECH \times VB$ condition and performed 3 practice trial series per $TECH \times VB \times MM$ condition.

Experiment 3: Results and Discussion

As in Experiments 1 and 2, we perform analyses of variances with the full factorial model $VB \times TECH \times MM \times WF \times Random(PARTICIPANT)$ for MT , FTT , CPT and ER . Tukey post-hoc tests are used for pairwise comparisons.

As expected, we find a simple effect of VB on MT ($F_{2,22} =$

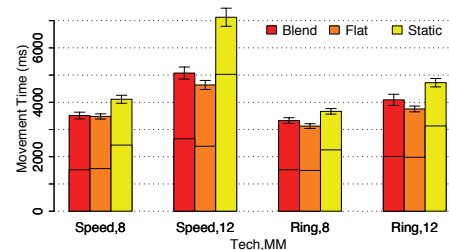


Figure 12. Movement time (MT) per VB by $TECH \times MM$ condition. The lower part of each bar represents focus targeting time (FTT), the upper part cursor pointing time (CPT).

67, $p < 0.0001$) revealing that visual behaviors significantly improve overall performance. Even if CPT is significantly degraded, the gain in FTT is strong enough (significantly) to decrease MT (see Figure 12). The degraded cursor pointing performance observed here is not surprising. It can be explained by the time it takes for a speed-coupled blending lens to become opaque enough or for a speed-coupled flattening lens to revert to its actual magnification factor. The performance gain measured for the focus targeting phase is consistent with previous experimental results [12, 18]. Overall, the gain in the focus targeting phase is strong enough to improve overall task performance.

The effects of WF and MM on MT are consistent with the previous two experiments: MT increases as WF decreases and as MM increases. *Ring* is still significantly faster than *Speed* ($TECH$ has a significant effect on MT : $F_{1,11} = 153, p < 0.0001$). Even if visual speed-coupling improves the performance of *Speed* more than that of *Ring* (significant interaction effect of $TECH \times VB$ on MT : $F_{1,11} = 11, p = 0.0005$), *Ring* remains faster than *Speed* for each MM . However, the advantage of *Ring* over *Speed* is significant only for $MM=12$ when we consider only the two speed-coupling techniques ($TECH \times MM$ on MT is significant, $F_{1,11} = 227, p < 0.0001$, as well as $VB \times TECH \times MM$, $F_{2,22} = 21, p < 0.0001$).

CHI 2010: Interfaces and Visualization

April 10–15, 2010, Atlanta, GA, USA

Note that we do not observe a significant advantage of *Blend* over *Flat* as reported in [18]. The main difference is that our targets are much smaller than those tested with Sigma lenses (0.25 to 1.9 context pixels in our experiment vs. 8 context pixels in [18]). Small targets probably cause more overshoot errors that are more expensive to repair with *Blend* than with *Flat*: if the larger flat-top of *Blend* is supposed to make focus targeting easier under an error-free hypothesis, it also causes an area of occlusion that is a significant drawback when trying to correct overshoots. Our participants actually reported that observation; in case of an overshoot they often left the target zone completely to perform a new focus targeting task. However this interpretation should be taken carefully since we did not record the number of overshoot errors. We only measured *ER*, the percentage of clicks outside the target (5.15% for *Blend*, 5.55% for *Flat* and 4.36% for *Static*). As in Experiment 2, the only factor that has an effect on error rate is target width *WF*.

SUMMARY AND FUTURE WORK

Large differences in scale between focus and context views cause a quantization problem that makes it difficult to precisely position lenses and to acquire small targets. Quantization severely limits the range of magnification factors that can be used in practice. We have introduced three high-precision techniques that address this problem, making focus targeting and object selection more efficient while allowing for higher magnification factors than regular lenses. This is confirmed by the results of our evaluations, which also reveal that some lenses are more robust than others for extreme conditions, with the *Ring* technique performing the best. Our high-precision techniques can be made even more efficient by combining them with speed-dependent visual behaviors drawn from the Sigma lens framework, as shown in the last experiment.

We analyzed our observations based on a model for target acquisition that sums the *focus targeting* and *cursor pointing* time to get the overall task time. Our results suggest that this model is too simple as it ignores the transition period between the two subtasks. This is especially true for lenses with a speed-dependent behavior, because of the delay to revert back to their stationary configuration. As future work we plan to refine the additive model to better account for these transitions. We also plan to adapt our techniques to other focus+context interfaces and investigate non-circular focus shapes.

REFERENCES

1. Y. Ayatsuka, J. Rekimoto, and S. Matsuoka. Popup vernier: a tool for sub-pixel-pitch dragging with smooth mode transition. In *Proc. UIST '98*, 39–48. ACM, 1998.
2. R. Balakrishnan. "Beating" Fitts' law: virtual enhancements for pointing facilitation. *IJHCS*, 61(6):857–874, 2004.
3. E. A. Bier, M. C. Stone, K. Pier, W. Buxton, and T. D. DeRose. Toolglass and magic lenses: the see-through interface. In *Proc. SIGGRAPH '93*, 73–80. ACM, 1993.
4. S. Carpendale, J. Ligh, and E. Pattison. Achieving higher magnification in context. In *Proc. UIST '04*, 71–80. ACM, 2004.
5. O. Chapuis, J. Labrune, and E. Pietriga. Dynaspot: Speed-dependent area cursor. In *Proc. CHI '09*, 1391–1400. ACM, 2009.
6. A. Cockburn and P. Brock. Human on-line response to visual and motor target expansion. In *Proc. GI '06*, 81–87, 2006.
7. A. Cockburn and A. Firth. Improving the acquisition of small targets. In *Proc. BCS-HCI '03*, 181–196, 2003.
8. A. Cockburn, A. Karlson, and B. B. Bederson. A review of overview+detail, zooming, and focus+context interfaces. *ACM CSUR*, 41(1):1–31, 2008.
9. S. A. Douglas, A. E. Kirkpatrick, and I. S. MacKenzie. Testing pointing device performance and user assessment with the ISO 9241, part 9 standard. In *Proc. CHI '99*, 215–222. ACM, 1999.
10. G. Fitzmaurice, A. Khan, R. Pieké, B. Buxton, and G. Kurtenbach. Tracking menus. In *Proc. UIST '03*, 71–79. ACM, 2003.
11. G. W. Furnas and B. B. Bederson. Space-scale diagrams: understanding multiscale interfaces. In *Proc. CHI '95*, 234–241. ACM & Addison-Wesley, 1995.
12. C. Gutwin. Improving focus targeting in interactive fish-eye views. In *Proc. CHI '02*, 267–274. ACM, 2002.
13. K. Hornbæk, B. B. Bederson, and C. Plaisant. Navigation patterns and usability of zoomable user interfaces with and without an overview. *ACM ToCHI*, 9(4):362–389, 2002.
14. T. Igarashi and K. Hinckley. Speed-dependent automatic zooming for browsing large documents. In *Proc. UIST '00*, 139–148. ACM, 2000.
15. S. Jul and G. W. Furnas. Critical zones in desert fog: aids to multiscale navigation. In *Proc. UIST '98*, 97–106. ACM, 1998.
16. J. Lamping and R. Rao. Laying out and visualizing large trees using a hyperbolic space. In *Proc. UIST '94*, 13–14. ACM, 1994.
17. M. J. McGuffin and R. Balakrishnan. Fitts' law and expanding targets: Experimental studies and designs for user interfaces. *ACM ToCHI*, 12(4):388–422, 2005.
18. E. Pietriga and C. Appert. Sigma lenses: focus-context transitions combining space, time and translucence. In *Proc. CHI '08*, 1343–1352. ACM, 2008.
19. G. Ramos, A. Cockburn, R. Balakrishnan, and M. Beaudouin-Lafon. Pointing lenses: facilitating stylus input through visual-and motor-space magnification. In *Proc. CHI '07*, 757–766. ACM, 2007.
20. M. Sarkar, S. S. Snibbe, O. J. Tversky, and S. P. Reiss. Stretching the rubber sheet: a metaphor for viewing large layouts on small screens. In *Proc. UIST '93*, 81–91. ACM, 1993.
21. J. J. van Wijk and W. A. Nuij. A model for smooth viewing and navigation of large 2d information spaces. *IEEE TVCG*, 10(4):447–458, 2004.
22. C. Ware and M. Lewis. The DragMag image magnifier. In *Proc. CHI '95*, 407–408. ACM, 1995.

Exploratory Analysis of Time-Series with ChronoLenses

Jian Zhao, Fanny Chevalier, Emmanuel Pietriga, and Ravin Balakrishnan



Fig. 1. The ChronoLenses interface includes (A) a charts panel showing the time-series; (B) a lens creation toolbar; (C) a lens analysis pipeline view of (D) the currently selected lens; (E) a property panel showing details of the currently selected lens. A context menu (F) can be invoked to perform lens-based operations, and (G) the lens toolbar allows quick access to a lens' parameters.

Abstract—Visual representations of time-series are useful for tasks such as identifying trends, patterns and anomalies in the data. Many techniques have been devised to make these visual representations more scalable, enabling the simultaneous display of multiple variables, as well as the multi-scale display of time-series of very high resolution or that span long time periods. There has been comparatively little research on how to support the more elaborate tasks associated with the exploratory visual analysis of time-series, e.g., visualizing derived values, identifying correlations, or discovering anomalies beyond obvious outliers. Such tasks typically require deriving new time-series from the original data, trying different functions and parameters in an iterative manner. We introduce a novel visualization technique called ChronoLenses, aimed at supporting users in such exploratory tasks. ChronoLenses perform on-the-fly transformation of the data points in their focus area, tightly integrating visual analysis with user actions, and enabling the progressive construction of advanced visual analysis pipelines.

Index Terms—Time-series Data, Exploratory Visualization, Focus+Context, Lens, Interaction Techniques.

1 INTRODUCTION

Time-series data are found in almost every domain, ranging from finance to many engineering and scientific disciplines. Time has inherently unique characteristics: for most purposes outside the theory of relativity, things evolve over time, but time does not depend on other variables. Time is considered uniform and absolute. It is thus often treated as a special variable, in terms of both how the data is structured and how it is presented to users. Tasks associated with the analysis of temporal datasets typically focus on the evolution of other

(dependent) variables with respect to time: identifying trends and recurring patterns, establishing correlations and possibly predicting the future based on past and current behavior.

Visual representations of time-series take advantage of people's innate perceptual abilities to process information and detect structure [32], making it significantly easier for users to discover trends and patterns at different scales, but also to identify anomalies in the data [10, 31]. However, basic time-series visualizations using line plots do not scale well; and as time-series data are often very large, featuring multiple, possibly heterogeneous, dependent variables measured for long periods of time and/or at high sampling rates, visualization of real-world time-series data poses significant challenges and has been an active area of research for many years. Many interactive visualization tools have been developed to address this scalability problem, offering innovative alternatives to the common line plot visualizations or enhancing the visualization with advanced interactive features.

There has been comparatively little research on how to support the more elaborate tasks typically associated with the exploratory visual analysis of time-series, e.g., visualizing derived values, identifying correlations, or identifying anomalies beyond obvious outliers. Such tasks typically require deriving new time-series from the data, visualizing those time-series and relating them to the original data plots.

- Jian Zhao is with DGP, University of Toronto, Canada. E-mail: jianzhao@dgp.toronto.edu.
- Fanny Chevalier is with OCAD University, Canada. E-mail: fchevalier@ocad.ca.
- Emmanuel Pietriga is with INRIA, France. E-mail: emmanuel.pietriga@inria.fr.
- Ravin Balakrishnan is with DGP, University of Toronto, Canada. E-mail: ravin@dgp.toronto.edu.

Manuscript received 31 March 2011; accepted 1 August 2011; posted online 23 October 2011; mailed on 14 October 2011.

For information on obtaining reprints of this article, please send email to: tvcg@computer.org.

Visual exploration techniques take advantage of human abilities to drive the data exploration process and are especially useful for undirected searches, when users know little about the data or have only vague exploration goals [23, 35]. As emphasized by Keim's visual analytics mantra – “*Analyze first, Show the important, Zoom, Filter and analyze further, Details on demand*” [24] – this process is iterative. For it to be efficient, visual representations, that support human judgment, and interactions, that re-parameterize the visual representation, should be tightly integrated, enabling users to quickly choose and refine parameter values that best suit the task at hand [2]. New plots derived from the original data should be put in context and made easy to relate both to the original data and to other plots that have been derived as part of the exploratory process.

We introduce a novel, domain-independent time-series visualization technique called ChronoLenses, aimed at supporting users in exploratory visual analysis tasks. ChronoLenses, as many other types of lenses, delimit a region of interest in the data to be put in focus through magnification [12, 33], visual filtering [15] or other arbitrary transformations of the underlying content [8]. Based on the metaphor of direct manipulation, ChronoLenses perform on-the-fly transformation of the data points in their focus area, tightly integrating visual analysis with interaction. Users can build pipelines composed of lenses performing various transformations on the data (e.g., *remove mean, compute 1st derivative, auto-correlation*), effectively creating flexible and reusable time-series visual analysis interfaces. At any moment, users can change the parameters of already created lenses, with the modifications instantaneously propagating down through the pipeline, providing immediate visual feedback that supports the iterative exploration process. Figure 1 gives an overview of the technique.

After a review of related work, we introduce tasks and requirements associated with the exploratory analysis of time-series, gathered from expert users in varied domains including operations monitoring and control for a radio-observatory, environmental research related to weather forecast, as well as financial and network streaming data analysis. We then describe the general concept of ChronoLenses, followed by its implementation in a research prototype application that provides rich interaction and analytical support for time-series. Example analysis pipelines follow with usage scenarios based on our work with network analysts, and astronomers of the ALMA radio-observatory, demonstrating how ChronoLenses can be used for the monitoring and offline analysis of complex time-series data. We conclude the paper with a discussion of the current implementation's limitations.

2 RELATED WORK

2.1 Time-series Visualization

Visual representations of time-series date back to the 18th century with seminal work using line charts by Playfair [37]. Many new static and dynamic techniques have been proposed since then. See [1, 32, 36] for relevant surveys. Some innovative techniques focus on helping users identify periodic patterns in the data. For instance, van Wijk [20] uses a calendar-based display to visualize time-oriented series aggregated on a daily, weekly or monthly basis via similarity clustering. Other visualizations, such as SpiralGraph [40] and SpiralView [7], lay out the data on a spiral-shaped time axis to reveal cycles. These techniques can yield meaningful representations of data that feature periodicity, making recurring patterns easy to spot. VizTree [28] takes a different approach, computing a symbolic representation of time-series and then representing the sequence of symbols using a suffix tree. The resulting visualization is radically different from other temporal visualizations and can be disconcerting at first, but represents a potentially powerful alternative to other techniques for identifying patterns in large datasets.

Most of the work on time-series visualization has focused on the scalability of the more conventional line plot and bar chart representations, either by proposing variations on the original plotting techniques or by enhancing them with advanced interactive visual filtering techniques. Variations on conventional line plots include Small multiples and sparklines [37], Horizon graphs [18], and Braided graphs [22]. These techniques focus on the issue of optimal space management, enabling the display of an increased number of time-series compared to

regular line plots. Lopez-Hernandez *et al.* [29] address the same problem for a specific type of time-series – univariate oscilloscope digital signals – by wrapping the signal in time, representing the different traces on layers that can be brushed by users.

2.2 Multi-scale Representations

Multi-scale visualization techniques adapt the representation depending on available screen real-estate. Time-series can be represented at several levels of detail and abstraction using different qualitative scales, or hybrids combining qualitative and quantitative information [6], gradually revealing more information as more space gets allocated to the chart, or as the user zooms in a region of interest [27]. Line Graph Explorer [25] provides an overview+detail interface for the exploration of large collections of time-series. It displays a compact overview of the entire collection by encoding the y-dimension of individual line graphs using color instead of space and viewing selected graphs in detail as standard line graphs. The technique is also interesting as the overview visualization lends itself well to sorting and clustering of the graphs using various similarity distances.

Hao *et al.* [17] present a space-filling, multi-resolution matrix representation of time-series where the color of a cell encodes the magnitude of the corresponding value in the data. The same authors propose another space-filling visualization capable of displaying multiple time-series, where the various charts are organized based on their importance in a treemap-like structure [16]. Stack zooming [21] uses a relatively similar technique, taking advantage of the one-dimensional nature of time-series to present them as hierarchies of strips, where each strip contains magnified versions of one or more region(s) of interest delimited in the above strip. Stack zooming can also be seen as a specialization of the DragMag [39] that supports recursive magnification and automatic arrangement of magnification windows. The technique enables efficient exploration of time-series at multiple levels of detail. LiveRAC [31] offers another variation on this approach, organizing charts in a reorderable spreadsheet-like matrix that allows side-by-side visual comparisons at multiple levels of detail. The technique supports semantic zooming [12], providing visual representations adapted (cell-wise) to the allocated screen space.

2.3 Lens-based Interaction for Time-series Visualizations

Timeboxes [19] take a different approach to the scalability problem. Timeboxes act like visual filters that specify constraints on what time-series to display, only showing plots that intersect the one or more boxes that form conjunctive queries, enabling the dynamic exploration of large data sets by direct manipulation of the rectangular boxes. In that sense, timeboxes can be seen as data filtering lenses, that have an impact on the visualization beyond the region of interest.

SignalLenses [26] provide another type of lens, conceptually closer to the usual lens-based focus+context visualization techniques [12]. SignalLenses are used for the visual analysis of large electronic time-series and help perform tasks such as anomaly detection and motif discovery. They provide in-place magnification of the signal, achieving a smooth transition between the focus and context regions through 1D distortion [11, 33]. Though they can provide additional measurement tracks to assist the exploration of time-series data by computing time-aligned properties, SignalLenses are limited to the magnification of plots in the region of interest. The MagicAnalytics lenses of KroNoMiner [41] go beyond simple magnification, building on the concept of Magic Lenses [8] originally designed for arbitrary 2D graphics. MagicAnalytics lenses compute and display the result of a function involving two time-series. They represent one of the basic building blocks of ChronoLenses, though limited to single step transformations.

ChronoLenses go beyond these simple transformations, offering multi-step transformations of one or more time-series. These can be purely visual transformations, or transformations that change the value of the data (1st derivative, point-wise maximum, etc.). ChronoLenses enable users to construct elaborate visual analysis pipelines, with modifications automatically propagating downstream, providing immediate visual feedback that supports the iterative exploration process.

3 TASKS AND DESIGN REQUIREMENTS

To better understand the needs for interactive visualization when exploring time-series data, we conducted informal interviews with expert users from various domains, including: astronomers, experts doing research in network streaming data analysis, finance, and weather forecasting. Based on their feedback and on the literature [4], we identified low-level tasks that users typically perform to carry out time-series analyses. From these tasks, we derived design requirements for the ChronoLenses technique, as introduced in this section.

The analysis of time-series datasets typically implies feature extraction and data comparison by transforming one or more time-series into another. Such computations usually require performing one, or a combination of, the following low-level tasks:

T1 Single-data stream transformation. Transform each data point of a series by applying an operator. Examples are data alteration, e.g., Fourier transforms and Box-Cox transforms [9]; bias reduction, e.g., remove means and remove trends; repeated pattern identification, e.g., auto-correlation; and operations related to delays and time lags, e.g., differencing and seasonal differencing.

T2 Cross-data stream analysis. Compute a new time-series from two or more input time-series. Examples are data comparison, e.g., subtraction; similarity examination, e.g., inner product; relationship discovery, e.g., cross-correlation; and series aggregation, e.g., point-wise maximum value.

However, exploring time-series often implies going through a more elaborate analysis pipeline that combines various tasks [24]. Practically speaking, this implies that users should be able to make compound queries on the data by iteratively performing sequences of low-level tasks **T1** and **T2**, combining and modifying the transformation parameters as part of the visual exploration process. From these observations we derive the following requirements:

R1 Dynamic transformations. Low-level tasks **T1** and **T2** are the core components of the visual analysis process. The transformations that support them should be easy to perform through operators applied to the input time-series data. Visual representations and interactions that re-parameterize these transformations should be tightly integrated so as to facilitate exploratory analysis.

- (a) *Dynamic selection of input region of interest:* users should be able to dynamically select and modify what timespan(s) in the input data are to be processed through the operators.
- (b) *Dynamic transformation parameters:* users should be able to easily configure and edit transformation parameters.
- (c) *Immediate visual feedback:* the system should provide instant visual update to reflect these changes.

R2 Dynamic analysis pipeline. Enabling the easy combination of operators makes it possible to progressively build and refine the analysis pipeline, thereby helping formulate complex queries [24].

- (a) *Dynamic composition:* users should be able to build the analysis pipeline iteratively, combining basic transformations through the incremental composition of operators that take as input arbitrary combinations of time-series from the original dataset and time-series that result from earlier transformation steps upstream in the pipeline.
- (b) *Reuse of intermediate results and easy backtracking:* The above requirement entails that all intermediate time-series transformation steps in the pipeline should be reusable as input to downstream operators, enabling users to branch out and explore, and possibly compare, alternatives at any point while sharing the data transformation steps that come earlier in the analysis process.
- (c) *Visual representation of the pipeline:* the system should provide an overview of the analysis pipeline, helping users maintain a mental map of the transformation steps and keep track of the exploration history.

We also rely on general principles for visual exploration as listed in [41], including: direct manipulation; overview first, zoom & filter, details on demand; and support for dynamic multi-focus exploration.

4 CHRONOLENSES FRAMEWORK

ChronoLenses is an interactive visualization technique for the exploratory analysis of time-series data, whose design was driven by the above requirements. It computes on-the-fly transformations of data points and displays the result of those transformations in place, using the metaphor of lenses. The MagicAnalyticsLens technique introduced in [41] relies on the Magic Lens concept [8] and forms one of the basic building blocks for ChronoLenses. While MagicAnalyticsLenses enabled users to apply four single-step basic transformations to exactly two input time-series, ChronoLenses extends the concept, allowing for an arbitrary number of input time-series and introducing several additional operators. But most importantly ChronoLenses enable multi-step transformations to be specified, where the result of time-series transformed through a given lens can serve as input to another lens, and where multiple pipelines can be branched out to explore multiple alternative visualizations simultaneously, easing the formulation of complex queries.

4.1 Lens Parameters

We define a lens $\mathcal{L}$ as the transformation of an input time-series in the focus region of that lens into a resulting time-series. Time-series can be seen as streams of data that get transformed through the lenses that constitute an analysis pipeline structured as a dataflow. Transformations are computed on-the-fly according to a set of parameters that can be dynamically adjusted¹. To support tasks **T1** and **T2**, $\mathcal{L}$ can either transform a single data stream, or perform cross-data stream operations. Data streams can be univariate or multivariate. Each lens $\mathcal{L}$ is defined by four transformations, all optional:

Unary Operator: $\mathcal{L}_{\text{unary}}(\cdot)$ defines the transformation that applies to a single data stream in the focus region of the lens (**T1**);

Binary Operator: $\mathcal{L}_{\text{binary}}(\cdot, \cdot)$ defines the transformation that takes the data in the focus region of the lens (processed by $\mathcal{L}_{\text{unary}}$ if set to anything else than the identity transform) as the first operand, and the output data stream resulting from the parent lens in the hierarchy (detailed later), if any, as the second² operand (**T2**);

Filter: $\mathcal{L}_{\text{filter}}(\cdot, \theta)$ defines visual filters that hide time-series specified by parameter θ (applies to multivariate data streams only);

Scaling: $\mathcal{L}_{\text{scale}}(\cdot, s)$ determines the magnification factor s applied to the data rendered in the lens' focus.

Operators $\mathcal{L}_{\text{unary}}$ and $\mathcal{L}_{\text{binary}}$ perform actual computations on the input data and can take some data points outside the lens' time span, such as when computing the 1st derivative. On the contrary, $\mathcal{L}_{\text{filter}}$ and $\mathcal{L}_{\text{scale}}$ only affect the visual representation of the processed data within that time span. They do not need to access data outside it.

4.2 Pipelines of Lenses

As mentioned earlier, the output of a lens, i.e., the time-series resulting from the transformation of an input time-series, can be fed to one or more lenses. In other words, lenses can be piped, effectively creating a lens-based data flow pipeline that can be used to progressively build elaborate visual analysis interfaces. The hierarchical combination of lenses relies both on a layering system and on cross-data stream parent-child relationships between lenses. The tree structure not only helps users keep track of the sequence of exploration steps, but also makes it possible to *backtrack*, that is, adjust the intermediate processing stages iteratively, the system providing immediate visual feedback

¹ChronoLenses support the representation of stacked multivariate time-series. In that case, each individual series is processed separately using the $\mathcal{L}$ transformation, and is then stacked in the lens' frame.

²Most of the operations that consider more than two input data streams can usually be simulated by cascading a series of binary operations.

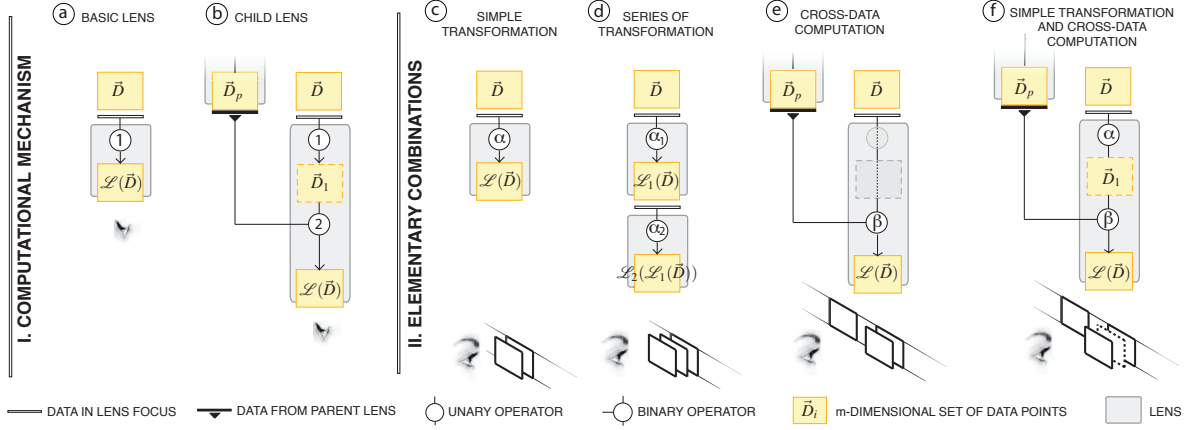


Fig. 3. Schematic representation of the ChronoLenses combination mechanism (I), and all possible elementary combinations (II).

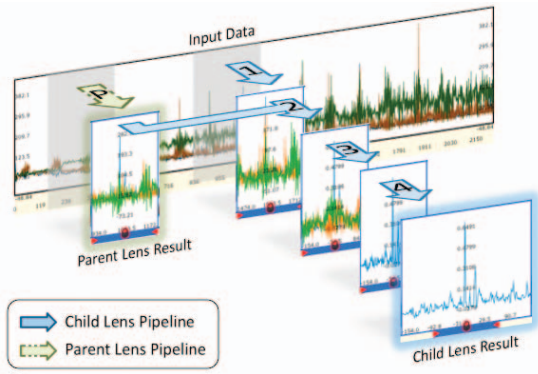


Fig. 2. ChronoLenses rendering pipeline for lens $\mathcal{L}$. (P) processing pipeline in parent lens; (1) applying Unary Operator $\mathcal{L}_{unary}$; (2) applying Binary Operator $\mathcal{L}_{binary}$; (3) applying Filter Operator $\mathcal{L}_{filter}$; (4) applying Scaling Operator $\mathcal{L}_{scale}$.

of the consequences of these parameter adjustments by propagating them downwards in the hierarchy. In the following, we describe the steps involved in rendering data seen through a lens, and then describe how lenses can be combined to build a full analysis pipeline.

4.2.1 Rendering Pipeline

Figure 2 illustrates the steps of the rendering pipeline applied to some multivariate input data, eventually leading to the visualization of the resulting time-series through lens $\mathcal{L}$, that gets part of its input from a parent lens.

More formally, the rendering pipeline can be described as follows. Let $\vec{D} = (d_1, d_2, \dots, d_m)^T$ be the m -dimensional set of data points defined by the focus region of lens $\mathcal{L}$. We note $\mathcal{L}_p(\vec{D}_i)$ the result of applying one of the transformations steps to the data, where $op \in \{unary, binary, filter, scale\}$. The final result $\mathcal{L}(\vec{D})$ is obtained by performing the following four computational steps:

- C1** Apply unary operator $\mathcal{L}_{unary}$, such that $\vec{D}_1 = \mathcal{L}_{unary}(\vec{D})$ (Figure 2-1);
- C2** Apply binary operator $\mathcal{L}_{binary}$, such that $\vec{D}_2 = \mathcal{L}_{binary}(\vec{D}_1, \vec{D}_p)$ (Figure 2-2), where $\vec{D}_p$ is the set of data points resulting from applying the parent lens to the same or to another set of input data points (Figure 2-P);

C3 Apply visual filter operator $\mathcal{L}_{filter}$ such that $\vec{D}_3 = \mathcal{L}_{filter}(\vec{D}_2, \theta)$, where θ defines the subset of time-dependent variables to be rendered in case of multivariate input (Figure 2-3);

C4 Apply scaling operator $\mathcal{L}_{scale}$ such that $\vec{D}_4 = \mathcal{L}_{scale}(\vec{D}_3, s)$, where s defines the magnification factor (Figure 2-4). We note $\mathcal{L}(\vec{D}) = \vec{D}_4$ the final result.

Any of the four operators can be set to *null*, in which case the associated step is equivalent to an identity transform. As mentioned earlier, operators $\mathcal{L}_{unary}$ and $\mathcal{L}_{binary}$ perform actual computations on the input data while $\mathcal{L}_{filter}$ and $\mathcal{L}_{scale}$ only affect the visual representation of the data. Thus, a simple magnifying lens can be obtained by only setting $\mathcal{L}_{scale}$. And a lens showing only one time-series in a multivariate plot can be obtained by only setting $\mathcal{L}_{filter}$ with θ limited to that particular series. The following section focuses on the more complex computational steps C1 and C2, and describes complete analysis pipelines involving multiple lenses.

4.2.2 Analysis pipeline

Users can specify elaborate transformations by combining multiple lenses. Two lenses $\mathcal{L}_1$ and $\mathcal{L}_2$ can be combined either by overlaying them in the chart, in which case the operators defining the lenses get applied sequentially to the input time-series data points (first those of $\mathcal{L}_1$, then those of $\mathcal{L}_2$); or by declaring $\mathcal{L}_1$ as the *parent* lens of $\mathcal{L}_2$, the transformed output of $\mathcal{L}_1$ being an operand of $\mathcal{L}_2$'s binary operator. Figure 3.I gives a schematic representation of these two combination mechanisms. Basic transformations that apply to a single data stream only require a lens with a unary operator (Figure 3-a) as defined in C1. Cross-data-stream computations require the definition of a child lens (Figure 3-b), that takes as input a data stream resulting from a parent lens and the data in the child lens' focus. The two streams are processed through the child lens' binary operator (C2), with the unary operator (C1) pre-processing the data points in the child lens' focus, if set. Figure 3.II illustrates all possible elementary lens combinations:

- E1** *Simple transformation* (Figure 3-c): the most basic lens, where $\mathcal{L}_{unary}$ is set and $\mathcal{L}_{binary}$ is *null*. It is conceptually similar to a Magic Lens [8], with the unary operator (noted α) transforming the data in the lens' focus region;
- E2** *Series of transformations* (Figure 3-d): two lenses that are piled up (they observe the same timespan) on separate layers as in traditional graphics editors. The first lens processes the input time-series data points that fall into its focus through unary operator α_1 . The resulting data is fed to the second lens and gets processed by unary operator α_2 . It is possible to pile up an arbitrary number of lenses, as with Fishkin and Stone's Movable Filters [14].

- E3** *Cross-data computation* (Figure 3-e): cross-data computations are achieved by creating a parent-child relationship between two lenses. Output data from the parent lens and time-series data points in the child lens' focus region are the operands given to binary operator β . The unary operator of the child lens is set to *null* (identity transform).
- E4** *Simple transformation and cross-data computation* (Figure 3-f): both the unary operator α and the binary operator β are set. The lens first applies transformation α to the time-series data points in its focus. The result of this transformation, $\vec{D}_1$, is then fed to the binary operator β along with the output of the parent lens' transformation, $\vec{D}_P$.

More elaborate analysis pipelines can be built by creating hierarchies consisting of the above elementary combinations (see Section 6).

5 CHRONOLENSES INTERFACE

We developed a proof-of-concept, full-featured interactive visualization tool implementing the ChronoLenses technique for the visual exploration of multivariate time-series. The interface (Figure 1) consists of five main components: (A) the *chart panel* displaying the different time-series in a classical stacked view; (B) a *lens creation toolbar*; (C) a *lens analysis pipeline view* displaying all ancestors of the currently selected lens (D); and (E) a *property panel* showing the latter lens' settings. Following the design requirements listed in Section 3, the interactive visualization was designed to allow rapid exploration of time-series and construction of analytical pipelines. This section describes the interactive visual interface of our prototype. Concrete examples of use are introduced in the following section.

Figure 4-l shows the user interface of a lens $\mathcal{L}$. The region of interest (time span $\vec{D}$ in the lens focus) is visually represented as a blue *focus bar* on the time axis (Figure 4-c). The time-series $\mathcal{L}(\vec{D})$ is displayed inside the lens' frame (Figure 4-b). A toolbar located on the top border of the lens provides quick access to the lens' main parameters (Figure 4-a). This toolbar is visible only when the cursor is hovering over the lens, so as to minimize visual clutter. Detailed information of a lens is also provided in a tooltip when hovering.

The user can modify the visual representation of plots, offering different perspectives on the data. Our prototype supports classical plots such as line charts and dot plots, but also statistical plots including histograms and Q-Q normal plots for comparing data distributions.

5.1 Creating an Analysis Pipeline

Advanced analysis pipelines are typically built by creating and progressively combining multiple lenses corresponding to the desired elementary operations **E1-E4** defined in Section 4.2.2.

To create a basic lens, the user first specifies its parameters through the creation toolbar, for instance choosing a *unary operator* such as *remove means*, and setting the *binary operator* to *null*. She then selects the data to be processed through the lens by initiating a rubber-band selection on any time-series loaded in the charts panel. This selection, achieved thanks to a simple mouse drag (Figure 4-e), defines the time span of interest, that will become the lens' focus.

Lenses that enable cross-data stream analyses obviously accept two input streams³. One stream is defined by the lens' focus, set as explained above. The second stream is provided as the output of another lens. This lens is considered as the *parent lens*; it feeds data to the *child lens*, that performs the cross-data stream analysis. The parent lens must exist in order to create a child lens.

The user can derive a child lens from the currently selected lens, by clicking on the *Create Child Lens* button in the creation toolbar or on the equivalent contextual menu item (Figure 1-f), after having set the desired parameters for the *unary operator* (if relevant) and *binary*

operator (e.g. *cross-correlation*). The time span of a child lens must match that of the parent lens. This is to guarantee that input streams contain the same number of data points for binary operations, and thus make sure that cross-data stream operations are consistent. However, it would be possible to support unequal time spans between parent and child lenses, using time-series transformations that would make the number of data points between both streams consistent using, e.g., linear interpolation methods for time-series.

The created lenses are z-ordered using layers and can be piled up according to this ordering. Piling up lenses entails piping their operators as described in Section 4.2.2. In the current implementation, two lenses get actually piped when the upper lens' time span is fully contained within that of the lens underneath: the result of the latter then becomes the input of the former, as when composing Movable Filters by overlapping them [14].

To help the user keep awareness of the analysis pipeline, a pipeline view is provided, depicting the currently selected lens' ancestors (Figure 1-C). The tree structure displays dynamic miniatures of the different lens branches, alongside labels detailing the operators associated with each lens in the pipeline. The current lens is visually emphasized with a thick blue border, and always corresponds to the top-left element in the pipeline view. By hovering over a lens either in the pipeline view or in the main chart view, the user can get an overall preview of all its ancestors. All lenses upstream are visually identified using an outer-glowing effect varying from green (closest ancestor) to yellow (furthest ancestor) in both views. The hovered lens is also emphasized with a blue glow effect (Figure 1-D).

5.2 Visual Exploration through Direct Manipulation

Visual exploration, the process of interactively browsing through different regions of a dataset to gain a better understanding of it, is not only useful as a quick and effective technique for hypothesis confirmation. It is also, and perhaps more importantly, essential for discovering the unexpected and raising new questions [38]. Direct manipulation has been successfully applied to time-series interactive visualizations as a means to facilitate visual exploration in conjunction with immediate visual feedback; see, e.g., PatternFinder [10] and KronoMiner [41]. ChronoLenses, by virtue of their progressive analysis pipeline building process, are meant to facilitate step-by-step exploration through direct manipulation, while giving users freedom to edit intermediate steps. Changes to a lens in the hierarchy are automatically propagated to its descendants, and all affected plots are visually updated accordingly.

5.2.1 Modifying Lens Parameters

To facilitate opportunistic discovery of regions of interest, we make it possible to redefine the focus region of a lens by simply dragging the lens frame (Figure 4-g) or the associated focus bar (Figure 4-h) left or right. The immediate redisplay of the newly computed result in both the chart view and the pipeline view enables quick access to different regions of the data stream, and thereby effectively supports visual exploration. The red arrow controllers placed on the focus bar (Figure 4-f) make it possible to adjust the lens' time span. The time span of all lenses downstream is adjusted accordingly so as to keep the number of data points consistent, as explained earlier. The user can also decide to move a lens to a different chart to further her analysis on another dataset, as illustrated in Figure 4-k.

The lens frame's width can be increased using the mouse wheel. The time span considered does not change, which means that resizing affects the $\mathcal{L}_{scale}$ parameter: the lens behaves as a magnifying glass⁴. Resizing on the vertical axis works similarly, except that the lens frame's height never changes and is always equal to the underlying chart's bounding box height. The y-axis origin can be adjusted by dragging the plot up and down inside the frame. An auto-adjust function is also provided, that automatically adjusts the y-axis' origin and scale so as to make the best use of available screen real-estate within the lens' frame as the user drags it.

⁴The above-mentioned blue bar still shows the original region of interest.

³As explained in Section 4.2.1, these streams can contain multivariate time-series. The binary nature of the transformation does not mean that input time-series are restricted to two variables. See Section 5.3 for more information about handling multivariate time-series.

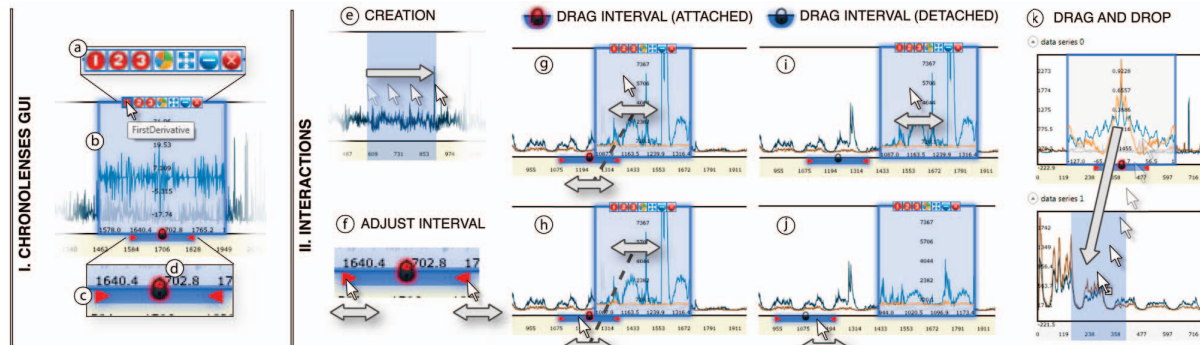


Fig. 4. (I) UI of a ChronoLens: (a) toolbar enabling adjustment of the lens' main parameters, (b) tooltip (pops-up when hovering), (c) focus bar with resizing handles, (d) lock icon for detaching/reattaching the lens' frame from/to the focus bar; (II) associated interactions.

5.2.2 Groups of Lenses

As mentioned earlier, the z-ordering of lenses that belong to the same analysis pipeline implicitly defines in what order transformation operations are applied to the data. The z-ordering and relative position of lenses that make a given analysis pipeline is important and should be kept constant during visual exploration via direct manipulation (when brushing the time-series plots with the lenses). Having to move each lens would be extremely cumbersome and would greatly impede the interactive visual exploration process. To avoid this, lenses can be grouped. All lenses that belong to the same group move synchronously as the user drags any one of them. The user can therefore create a compound lens set in which lenses are piled up in a specific order, thus creating a reusable analysis pipeline.

Grouping lenses can also be useful when analyzing multiple charts at a time, or when looking for seasonality. Such tasks require the interval between lenses (time lag) in the data stream to remain constant. By defining a group consisting of the different lenses to be moved synchronously, the user can keep the time lag between these constant.

A lens can be added to the currently selected group through the contextual menu (Figure 1-F). All lenses belonging to the same group as the currently selected one are outlined with a green stroke, giving feedback about the time-lag constraints that exist between those lenses. Group membership is also reflected in the tree view, where the group number is explicitly specified alongside the lens miniatures.

5.2.3 Dealing with Visual Clutter

The instantiation of many lenses can quickly lead to visual clutter. Lenses that are part of the same analysis pipeline will often overlay each other, either partially or fully occluding one another. This can be a problem when the user wants to visualize intermediate computational steps of the pipeline. Occlusion problems can also arise when magnification lenses (any lens with $\mathcal{L}_{scale} > 1$) look at time spans that are disjoint but close to one another.

To address these issues, we introduce a mechanism that enables the user to detach the lens frame from its associated focus bar, thereby giving her control over where to display the lens' output, independently of the actual input region of interest (focus bar location). Figures 4-g to 4-j illustrate this mechanism. To detach a lens from its focus bar, the user simply clicks on the lock icon (Figure 4-d) to unlock the lens from the focus bar. In that mode, dragging the lens frame does not affect the focus interval (Figure 4-i), making it possible to space out lenses or, on the contrary, move them closer to facilitate comparison. In the same manner, dragging the focus bar only affects the input time span, the lens frame remaining in place (Figure 4-j). Wherever a lens and its focus bar are positioned, locking them again will keep the position offset constant (Figure 4-g-h). When hovering over a lens frame or a focus bar, both are emphasized with a glow effect making it easier to identify what focus bar is associated with what lens, and conversely.

Other mechanisms that help manage visual clutter include *delete* and *minimize* buttons in the toolbar (Figure 4-a). The former deletes

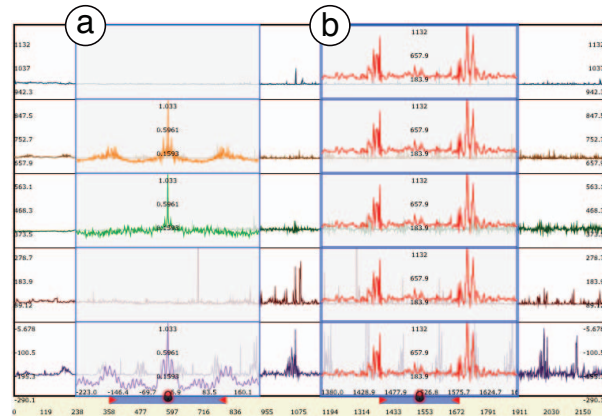


Fig. 5. Lenses applied to multivariate data: (a) auto-correlation for three series out of five; (b) point-wise aggregated maximum showing the resulting plot (common to all series) overlaid on top of each variable's plot.

the lens, the latter hides the lens' frame, but not its focus bar. The bar remains visible so as to keep the user aware of its existence. At all times, double-clicking it makes the lens' frame visible again.

5.2.4 Multi-foci Exploration

As many other types of datasets, time-series are amenable to multi-scale navigation. For a given series, the user might be interested in behaviors and event patterns that occur at various time scales, from months or years down to fractions of a second.

Magnification lenses support the process of interactively drilling down into the data, and enable the comparison of different time spans (a common practice when analyzing time-series data [13, 21, 41]) simply by instantiating multiple lenses. Magnification lenses can easily be instantiated in ChronoLenses: any lens with $\mathcal{L}_{scale} > 1$ is a magnification lens that enables drilling down into the data. However, magnification lenses pose some problems from an interaction perspective [5, 33]. In-place magnification of the focus region means that the immediate surroundings of the region of interest will be occluded if the lens is simply overlaid on the original data, hiding potentially valuable information and hindering navigation. Smooth transition between the magnified focus and surrounding context can be achieved using spatial distortion [26], but the introduced deformation has a cost in terms of legibility and interpretation of the visualization.

An alternative drill-down method called Stack Zooming was recently proposed by Javed *et al.* [21]. The technique consists in stacking multiple views as the user drills down into the data. The con-

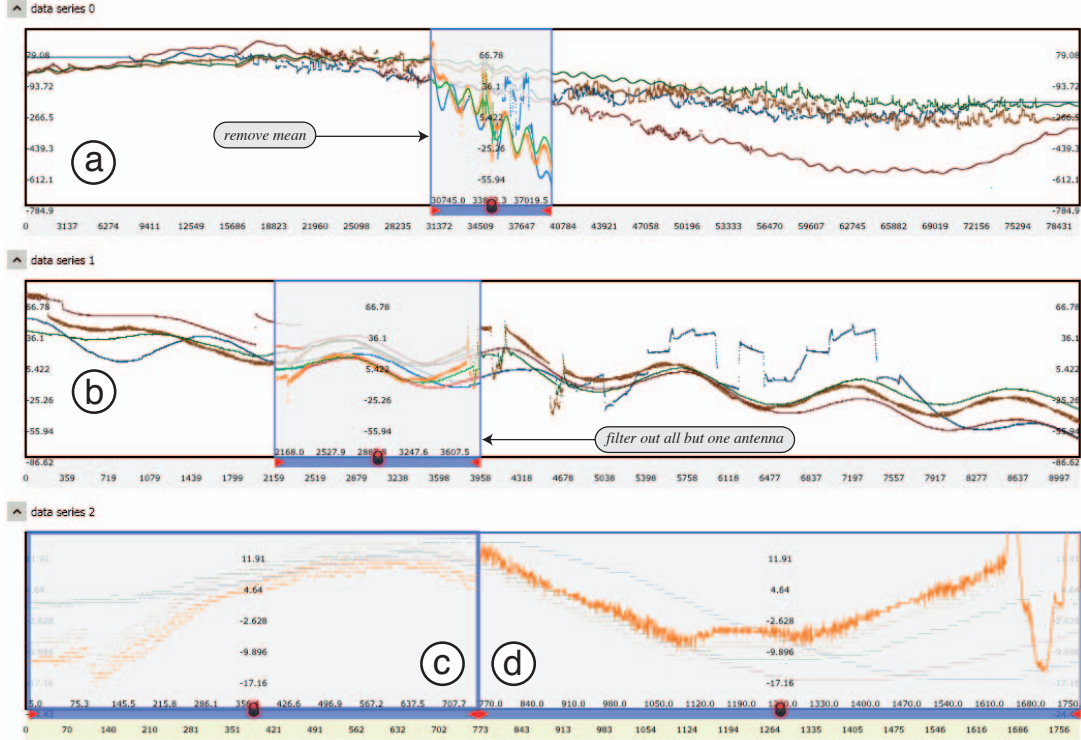


Fig. 6. Exploring ALMA Line Length Correction Stretcher Voltage plots for four antennas: (a) two-day overview at a sampling rate of one second; (b) magnification of the 5 hours seen through the *remove mean* lens applied in (a); magnification of the 50 minutes for a single antenna seen through a filtering lens, rendered in scatterplot mode (c) and line-plot mode (d).

tent of a given view is a magnification of a region in the view immediately above, delimited by a lens-like viewfinder similar to a (one-dimensional) DragMag [39]. The technique can seamlessly be integrated with ChronoLenses and is implemented as follows.

The user creates a new empty chart panel in the main view. Then, dragging-and-dropping an existing lens into this empty panel duplicates the data within the span of the lens and displays it in this new panel. The visual representation is stretched horizontally to fill the panel, thus providing a magnified view of the lens' content that is dynamically updated whenever that lens moves. Figure 6 illustrates this technique, with chart (b) providing a magnified version of the region seen through the lens in chart (a). Lenses can be instantiated on this new panel, enabling users to drill-down recursively and build a truly multi-scale view hierarchy in which lenses at any level can be freely adjusted. Lenses that form this hierarchy are of course not limited to simple magnification and can apply to the data any of the elaborate transformations enabled by the ChronoLenses operators.

5.3 Multivariate Time-Series

In ChronoLenses, multivariate datasets can be visualized either overlaid in the same chart (Figure 1, top chart) or separately, stacked on top of one another (Figure 1, bottom chart). When applying a lens to a multivariate data stream, each series gets processed independently, except for some operators that aggregate the data, such as *point-wise minimum* or *maximum*. In that case, the output is a univariate stream, that can be duplicated and overlaid in all charts. We distinguish aggregated operators from other operators by color-coding in red all the replicas of the unique plot resulting from data aggregation (Figure 5-b) as opposed to the one-to-one color mapping used when the different variables are processed separately (Figure 5-a).

To focus on a subset of variables, the user can filter out series that she does not want to be considered in the computational pipeline by setting the filter operator $\mathcal{L}_{filter}$ accordingly. To do so, she can either

select all the series to be taken into account in the corresponding menu accessible from the lens toolbar, or specify the list in the property panel (Figure 1-E), by entering the corresponding textual expression using a very simple syntax. When filtered out, streams are neither computed nor rendered, as the first and fourth streams in Figure 5-a. Note that when a lens defines an aggregated operator, filtering out a stream has an impact on the result, as the stream is no longer considered as an operand. Again, we offer the user full control over what data gets processed, making the analysis process highly flexible and customizable.

6 USE CASE SCENARIOS

In this section, we illustrate how ChronoLenses can be used with two use cases involving real datasets.

6.1 ALMA Observatory Usage Scenario

The Atacama Large Millimeter/submillimeter Array (ALMA, [3, 34]) is a single telescope (under construction in the Chilean Andes) that will eventually be composed of 66 high-precision antennas. Observations are based on the principle of interferometry: a source in the sky is observed by at least two antennas; the signals (radio waves) captured by each antenna are then combined by a central computer called the correlator to form images suitable for performing scientific analysis. Time-series visualization are used by both operators and astronomers for a variety of tasks, ranging from checking some of the thousands of monitor points in the system to performing scientific data quality assurance during observations. In the following, we focus on one example where ChronoLenses can help users in their daily task.

When combining the signals coming from the different antennas taking part in a given observation, the correlator must know the length of the path traveled by the signal through the fiber optics cables with an accuracy of hundredths of a millimeter. A round-trip laser signal gets sent to all antennas in order to continuously monitor the length of the optical fibers, as the latter can expand and contract due to temperature

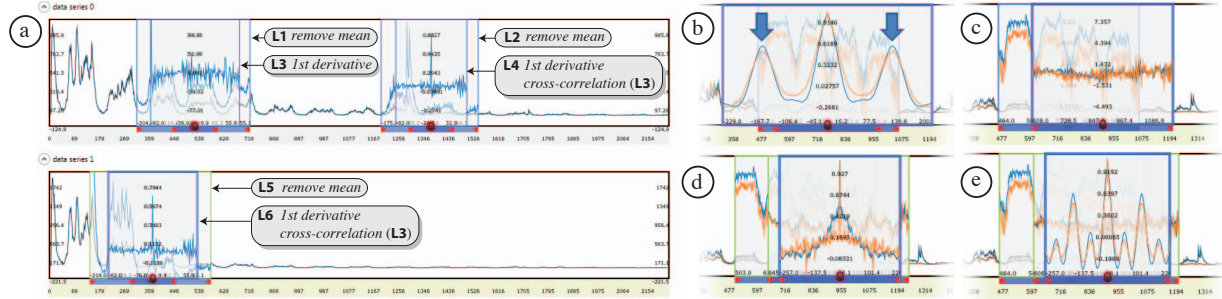


Fig. 7. Analysis of a month of the visiting population of two P2P video-on-demand channels: a) analysis of the evolution of the correlation within and between the channels (see Section 6.2.1); b) looking for seasonal pattern lag value, the blue arrows show a strong correlation at a delay of 1440s ($\approx 24h$) and c-d) comparing the delayed time spans (see Section 6.2.2).

variations. These changes in path length must be compensated in real-time. This is achieved by the Line Length Correction (LLC) system, which ensures that path lengths are all stabilized to about 1 micron. Operators are interested in monitoring the LLC stretcher voltage for each antenna, and observing potential deviation of an antenna's LLC stretcher voltage compared to that of the other antennas.

From a time-series visualization perspective, different things can be happening at different time scales: from several days to a few minutes. Efficient multi-scale and multi-focus visualization, as enabled by the $\mathcal{L}_{scale}$ operator and the variation on stack zooming [21] implemented in ChronoLenses, is thus an essential feature. Figure 6-a plots voltage against time for 2 days at a sampling rate of 1 second.

Starting from this 2-day overview, the operator is first interested in finding out whether all antennas are behaving normally or if one or more are somehow deviating. All antennas are expected to behave more or less similarly. Direct visual comparison between the plots, overlaid or stacked, is sufficient to see the deviation of one antenna during the second day (Figure 6-a).

To better see the smaller and shorter fluctuations, the operator creates a lens spanning a 5-hour period, with $\mathcal{L}_{unary}$ operator *remove mean* (Figure 6-a). The lens is dragged and dropped to create panel 6-b. The content of that panel is a stretched version of what is seen through 6-a's lens. The operator can clearly see that the fluctuations are in phase, which implies that they all come from a single source. She could then plot various monitoring points simultaneously based on system or environmental components likely to cause these fluctuations and create lenses defining a cross-correlation $\mathcal{L}_{binary}$ operator (not shown here), eventually tracing the source to temperature fluctuations in the local oscillator room.

Finally, zooming in further to display just 50 minutes (Figure 6-c), the operator can see some odd features in the signal for one of the antennas. She filters out antennas that behave normally using a lens defining the appropriate $\mathcal{L}_{filter}$. She also switches from a scatterplot to a line-plot rendering inside the lens focus (Figure 6-d), revealing fast oscillations at a much lower scale that occur on top of the slow ones observed earlier for this particular antenna. This antenna-specific issue eventually gets traced to a device repeatedly turning on and off in the antenna.

6.2 Peer-to-peer Network Fluctuation Analysis

We consider a network system manager who is trying to optimize bandwidth usage on a centralized peer-to-peer system. She uses line graphs of the visiting population of two different video-on-demand channels. The relative evolution through time of the two channels, temporal distances, and the activity load before and after peak events are important clues that the analyst can rely upon, assisting her in the decision making process. For instance, identifying regular activity patterns helps better predict future fluctuations and therefore better pinpoint the needs for server pool optimization. Being able to spot anomalies such as overloads in their context also helps identify the potential causes, and plan for technical solutions.

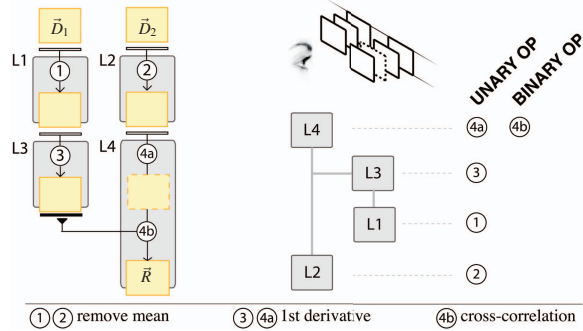


Fig. 8. Comparing the evolution of two time-series using ChronoLenses. Left: the analysis pipeline, right: the corresponding lens hierarchy.

6.2.1 Analyzing the Evolution of Correlations

Our dataset consists of the visiting population of two different channels over a month, sampled from the server every 10 minutes (Figure 7). The overall yearly or monthly trends of the channels might be different. For instance, one might be rising much faster than the other, or one might fall while the other is rising a bit. In this first scenario, the user is rather interested in the smaller scale fluctuations, comparing evolution and identifying potential correlations at this scale.

The user first focuses on a single channel, looking for repeating patterns. To do so, she compares differentiated data at different times by performing the following steps: 1) remove mean, to make the data more stationary and thus more amenable to comparison for our purposes; 2) compute the first derivative (or differencing) of those; and 3) perform the cross-correlation between the two for comparison. The pipeline that supports this type of analysis, described below, is composed of four lenses, as shown in Figure 8.

The user creates lenses L1 and L2, with the $\mathcal{L}_{unary}$ operator set to *remove mean* (step 1). She then creates L3 ($\mathcal{L}_{unary}$ operator set to *1st derivative*) and places it on top of L1 (step 2). Its child lens L4 gets positioned on top of L2 with operators $\mathcal{L}_{unary}$ and $\mathcal{L}_{binary}$ set to *1st derivative* and *cross-correlation* respectively (steps 2-3). In the end, L4 outputs the correlation of the data below it (1st derivative of L2) and its parent's output (L3). The discovery of recurring patterns usually requires performing computations on data with parameterized-yet-fixed lags in time. Here, we use lens groups – that make all member lenses move synchronously – to build dynamically-parameterized operators enabling efficient exploration of the effect of different time lags and time spans via direct manipulation.

Grouping L1 and L3 on the one hand (L1-3), and L2 and L4 on the other hand (L2-4) makes it possible to drag one or the other focused intervals along time and look for strong correlation by observing the

result of L4. When such a correlation is identified, as in Figure 7-a, the user can group L1-3 and L2-4 altogether and drag them. If L4 exhibits a stable plot while dragging, then a recurrent trend is identified. If on the contrary the plot varies, direct manipulation with immediate feedback helps better understand the correlation variation before and after peaks of interest, whereas it might be more difficult to apprehend when computing automatic pattern detection.

When the analyst has located a pattern of interest, she can focus on the second channel to look for correlations. She creates pipeline L5-6 similar to group L2-4 using the data from the second channel as input: L5 (remove mean) and L6 (L3's child placed on top of L5 with 1st derivative and cross-correlation operators), as depicted in Figure 7-a.

Different analysis tasks can easily be performed with the current set of lenses: (1) drag the group L5-6 until the plot is like, or the opposite of, L2-4 to identify correlations between the two series (see Figure 7-a); (2) group L2-4 and L5-6 so that they preserve their relative distance while dragging one or the other in order to compare the correlation fluctuations by simultaneously looking at the results of lenses L4 and L6. If those behave the same, then a similar evolution is identified and the relative lag is known. If they always behave the opposite, there is an inverse causality; and (3) by extending the group to L1-3 and dragging the whole, the analyst can also explore if a pattern is preserved in time. The user can also change the lenses' operators at any time during exploration, as for example changing L3, L4 and L6's $\mathcal{L}_{unary}$ operator to *2nd derivative* and looking for insights when more lag is involved (20 minutes in our example).

6.2.2 Comparing different Time Spans in the same Series

Discovering seasonality is another important task in time-series data analysis. It helps better predict future fluctuations. In this second part of the scenario, our user is interested in finding seasonal differencing on the two channels simultaneously, in order to gain a better understanding of what characterizes the typical data streaming cycle, if such a cycle exists. Gaining this knowledge will eventually help develop strategies to optimize bandwidth usage, but also assist in spotting abnormal behavior when it occurs.

Box-Cox Transforms [9] can be used to perform a variance stabilizing transformation. Seasonal differencing parameterized with a time lag can then be applied to find out if the non-stationarity of the original time-series is removed [30], but this requires to know the lag value in advance. The identification of an appropriate lag value to perform seasonal differencing can be made easier by building the following pipeline: an *auto-correlation* lens (L1) on top of a *Box-Cox transform* lens (L2). A strong correlation is observed at a delay of approximately 1440 minutes = 24 hours (see Figure 7-b).

The analyst deletes L2 and furthers her exploration as she creates L2's child lens L3 with *Box-Cox transform* and *subtraction*, synchronized so that L3 is always 24 hours ahead of its parent (Figure 7-c). An *auto-correlation* lens is then applied on top (after the differencing operation) to evaluate stability of the series. At this position in the original time-series (Figure 7-d), there does not seem to be any significant seasonal pattern, as high correlation values exist only at zero-delay. But moving the lens group along the timeline, one pattern is eventually discovered (Figure 7-e), calling for further exploration.

7 DISCUSSION

ChronoLenses aim at facilitating the creation of customized analysis pipelines for easy exploration and navigation through time-series datasets. Our initial prototype, although already offering a large set of functionalities, still has limitations. It could be improved in several ways, as discussed in this section.

7.1 Layering and Treeview Limitations

In its current implementation, the ChronoLenses interface does not support effective layer management for the z-ordering of lenses that pile up. Although the grouping of lenses keeps the z-ordering constant, when not grouped, selecting a lens puts it on the topmost layer. This might be annoying when dealing with complex pipelines as clicking on a lens only to get information about it or even by mistake can alter

the pipeline. Making layer management more stable and more easily configurable through a dedicated synchronized view would reduce the need for advanced planning when building the pipeline and would aid keep an accurate mental map of the data flow.

Similarly, the lens hierarchy treeview could be enhanced. As is, the hierarchical view dynamically changes as the user selects a different lens. This might be distracting and difficult to interpret. Moreover it is not interactive. There is an opportunity for improvement here, as an additional and complementary representation of the analysis pipeline could be used as an alternative means of performing exploration tasks if enriched with interactive capabilities. In particular, allowing for the creation of lenses, duplication of analysis branches, or other changes to the analysis pipeline from this alternate view would make it easier to build up and manage complex pipelines.

7.2 Integration of SigmaLens Focus+Context Techniques

Another limitation of the current implementation is related to the positioning of lenses. When displaying an overview of large time-series, ChronoLenses only draws a subset of all points (subsampling). Magnifying the data through a lens enables the display of more detail in context. However, as the lens' magnification factor increases, small movements of the lens translate to larger jumps in terms of time span observed in the lens' focus. For instance, given a magnification factor of 10x, moving the lens by 10 pixels will move the time span observed through the lens by 100 equivalent pixels at that zoom factor (see [5] for a detailed description of this problem of quantization). The problem also exists when controlling the time span seen in a plot through a lens observing a lower scale version of that plot à la Stack zooming. Adapting one of the high-precision magnification lens techniques introduced in [5] would solve this issue, enabling both fast repositioning and precise selection within the lens focus.

A related issue is that our lenses occlude the area immediately surrounding the region of interest, as basic magnifying lenses do. SigmaLenses [26] address this issue by distorting the representation in a bounded transition region between the lens focus and the context. We chose not to rely on spatial distortion because of the issues it raises in terms of making accurate analytical comparison and interpretation. However, there are other solutions to this problem, such as speed-coupled Sigma Lens focus targeting techniques [33]. For instance, the Speed-coupled Blending Lens technique consists in coupling the lens focus' opacity to its speed, smoothly fading out the lens as speed increases and smoothly fading it back in when it comes to a stop, having reached its target position. The technique addresses the problems of visual occlusion and would be relatively straightforward to implement in ChronoLenses as we already support translucency in lens renderings. However, potential issues related to visual interference between the layers during lens repositioning would have to be investigated.

8 CONCLUSIONS AND FUTURE WORK

We have presented ChronoLenses, a novel domain-independent visualization technique for the visual exploration of time-series data. ChronoLenses relies on the metaphor of lenses, that compute on-the-fly data transformations in place. ChronoLenses allow users to progressively build elaborate analysis pipelines by interactively compounding elementary operations, thus supporting complex user-defined exploratory analysis tasks.

The concept of ChronoLenses can be extended to other types of data than time-series, e.g., image analysis and image processing techniques for exploring temporal media such as video, where content-aware operators (feature detection algorithms, sharpening and color correction filters) would complement more generic operators (magnification, track filtering).

In addition to exploring new applications of the framework, we plan to improve our implementation by addressing the limitations discussed in the previous section, and by exploiting analytical mechanisms to guide exploration. For instance, lenses could be made to snap to local optima in the visualized time-series as the user drags them, using measures such as, e.g., the strongest local cross-correlation.

ACKNOWLEDGMENTS

We wish to thank all expert users that participated in our interviews and gave us feedback about the ChronoLenses tool, including: Denis Barkats from the Joint ALMA Office, National Radio Astronomy Observatory; Lindsey Davis from the National Radio Astronomy Observatory, Norbert Driedger and Brian Greaves from Environment Canada; Yashar Ganjali, from the Computer Systems and Networks Group at University of Toronto; Igor Jurisica from the Ontario Cancer Institute; Di Niu, from the Electrical and Computer Engineering department at University of Toronto; and Joseph Schwarz from the European Southern Observatory. We also thank Theophanis Tsandilas for helpful comments about early drafts of this paper.

REFERENCES

- [1] W. Aigner, S. Miksch, W. Müller, H. Schumann, and C. Tominski. Visualizing Time-oriented Data - a Systematic View. *Computers & Graphics*, 31(3):401–409, June 2007.
- [2] W. Aigner, S. Miksch, W. Müller, H. Schumann, and C. Tominski. Visual Methods for Analyzing Time-Oriented Data. *IEEE Transactions on Visualization and Computer Graphics*, 14(1):47–60, 2008.
- [3] ALMA. Atacama large millimeter/submillimeter array, Last accessed: June 2011. <http://www.almaobservatory.org>.
- [4] N. Andrienko and G. Andrienko. *Exploratory Analysis of Spatial and Temporal Data*. Springer, 2006.
- [5] C. Appert, O. Chapuis, and E. Pietriga. High-precision Magnification Lenses. In *Proceedings of the 28th international conference on Human factors in computing systems (CHI)*, pages 273–282. ACM, 2010.
- [6] R. Bade, S. Schlechtweg, and S. Miksch. Connecting Time-oriented Data and Information to a Coherent Interactive Visualization. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI)*, pages 105–112. ACM, 2004.
- [7] E. Bertini, P. Hertzog, and D. Lalanne. SpiralView: Towards Security Policies Assessment through Visual Correlation of Network Resources with Evolution of Alarms. In *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology (VAST)*, pages 139–146, 2007.
- [8] E. A. Bier, M. C. Stone, K. Pier, W. Buxton, and T. D. DeRose. Toolglass and Magic Lenses: the See-through Interface. In *Proceedings of the Conference on Computer graphics and interactive techniques (SIGGRAPH)*, pages 73–80. ACM, 1993.
- [9] G. E. P. Box and D. R. Cox. An Analysis of Transformations. *Journal of Royal Statistical Society*, B26:211–246, 1964.
- [10] P. Buono, A. Aris, C. Plaisant, A. Khella, and B. Shneiderman. Interactive Pattern Search in Time Series. In *Proceedings of Visualization and Data Analysis*, pages 175–186, 2005.
- [11] M. S. T. Carpendale and C. Montagnese. A Framework for Unifying Presentation Space. In *Proceedings of the 14th annual ACM symposium on User interface software and technology (UIST)*, pages 61–70. ACM, 2001.
- [12] A. Cockburn, A. Karlson, and B. B. Bederson. A Review of Overview+detail, Zooming, and Focus+context Interfaces. *ACM Computing Surveys (CSUR)*, 41:1–31, 2009.
- [13] N. Elmqvist, P. Dragicevic, and J. Fekete. Color Lens: Adaptive Color Scale Optimization for Visual Exploration. *IEEE Transactions on Visualization and Computer Graphics*, 2010. In press, Rapid post.
- [14] K. Fishkin and M. C. Stone. Enhanced dynamic queries via movable filters. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI)*, pages 415–420. ACM/Addison-Wesley Publishing Co., 1995.
- [15] G. W. Furnas. Generalized Fisheye Views. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI)*, pages 16–23. ACM, 1986.
- [16] M. C. Hao, U. Dayal, D. A. Keim, and T. Schreck. Importance-Driven Visualization Layouts for Large Time Series Data. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis)*, pages 203–210. IEEE Computer Society, 2005.
- [17] M. C. Hao, U. Dayal, D. A. Keim, and T. Schreck. Multi-Resolution Techniques for Visual Exploration of Large Time-Series Data. In *EuroVis*, pages 27–34, 2007.
- [18] J. Heer, N. Kong, and M. Agrawala. Sizing the Horizon: the Effects of Chart Size and Layering on the Graphical Perception of Time Series Visualizations. In *Proceedings of the 27th international conference on Human factors in computing systems (CHI)*, pages 1303–1312. ACM, 2009.
- [19] H. Hochheiser and B. Shneiderman. Dynamic Query Tools for Time Series Data sets: Timebox Widgets for Interactive Exploration. *Information Visualization*, 3(1):1–18, 2004.
- [20] J. J. Van Wijk. Cluster and Calendar based Visualization of Time Series Data. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis)*, pages 4–9, 1999.
- [21] W. Javed and N. Elmqvist. Stack Zooming for Multi-focus Interaction in Time-series Data Visualization. In *IEEE Pacific Visualization Symposium (PacificVis)*, pages 33–40, 2010.
- [22] W. Javed, B. McDonnell, and N. Elmqvist. Graphical Perception of Multiple Time Series. *IEEE Transactions on Visualization and Computer Graphics*, 16:927–934, 2010.
- [23] D. Keim. Information Visualization and Visual Data Mining. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):1–8, January 2002.
- [24] D. Keim, F. Mansmann, J. Schneidewind, and H. Ziegler. Challenges in Visual Data Analysis. In *Proceedings of the 10th International Conference on Information Visualization (IV)*, pages 9–16. IEEE, 2006.
- [25] R. Kincaid. Line Graph Explorer: Scalable Display of Line Graphs using Focus+context. In *In Working Conference on Advanced Visual interfaces (AVI)*, pages 404–411. ACM, 2006.
- [26] R. Kincaid. SignalLens: Focus+Context Applied to Electronic Time Series. *IEEE Transactions on Visualization and Computer Graphics*, 16:900–907, 2010.
- [27] H. Lam, T. Munzner, and R. Kincaid. Overview Use in Multiple Visual Information Resolution Interfaces. *IEEE Transactions on Visualization and Computer Graphics*, 13:1278–1285, 2007.
- [28] J. Lin, E. Keogh, S. Lonardi, J. P. Lankford, and D. M. Nystrom. Visually mining and monitoring massive time series. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 460–469. ACM, 2004.
- [29] R. Lopez-Hernandez, D. Guilmaine, M. McGuffin, and L. Barford. A Layer-oriented Interface for Visualizing Time-series Data from Oscilloscopes. In *IEEE Pacific Visualization Symposium (PacificVis)*, pages 41–48, 2010.
- [30] H. Madsen. *Time Series Analysis*. Chapman & Hall/CRC, Nov. 2007.
- [31] P. McLachlan, T. Munzner, E. Koutsosifos, and S. North. LiveRAC: Interactive Visual Exploration of System Management Time-series Data. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems (CHI)*, pages 1483–1492. ACM, 2008.
- [32] W. Müller and H. Schumann. Visualization methods for time-dependant data-an overview. In *Proceedings of the 2003 Winter Simulation Conference*, pages 737–745, 2003.
- [33] E. Pietriga, O. Bau, and C. Appert. Representation-Independent In-Place Magnification with Sigma Lenses. *IEEE Transactions on Visualization and Computer Graphics*, 16(3):455–467, 2010.
- [34] J. Schwarz, E. Pietriga, M. Schilling, and P. Grosbol. Goodbye to WIMPs: A Scalable Interface for ALMA Operations. In I. N. Evans, A. Accomazzi, D. J. Mink, and A. H. Rots, editors, *Astronomical Data Analysis Software and Systems (ADAS)*, ASP Conference Series. ASP, 2010.
- [35] B. Shneiderman. The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. *IEEE Symposium on Visual Languages (VL)*, pages 336–343, 1996.
- [36] S. F. Silva and T. Catarci. Visualization of Linear Time-Oriented Data: A Survey. In *Proceedings of the International Conference on Web Information Systems Engineering (WISE)*, pages 310–319. IEEE, 2000.
- [37] E. R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, May 2001. 197 pages.
- [38] J. W. Tukey. *Exploratory Data Analysis*. Addison-Wesley, 1977.
- [39] C. Ware and M. Lewis. The DragMag image magnifier. In *Conference companion on Human factors in computing systems (CHI)*, pages 407–408. ACM, 1995.
- [40] M. Weber, M. Alexa, and W. Müller. Visualizing Time-Series on Spirals. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis)*, pages 7–13, 2001.
- [41] J. Zhao, F. Chevalier, and R. Balakrishnan. KronoMiner: Using Multi-Foci Navigation for the Visual Exploration of Time-Series Data. In *Proceedings of the 29th international conference on Human factors in computing systems (CHI)*, pages 1737–1746. ACM, 2011.

CHI 2011 • Session: Mid-air Pointing & Gestures

May 7–12, 2011 • Vancouver, BC, Canada

Mid-air Pan-and-Zoom on Wall-sized Displays

Mathieu Nancel^{1,2} Julie Wagner^{2,1} Emmanuel Pietriga^{2,1} Olivier Chapuis^{1,2} Wendy Mackay^{2,1}

¹LRI - Univ Paris-Sud & CNRS
F-91405 Orsay, France

²INRIA
F-91405 Orsay, France

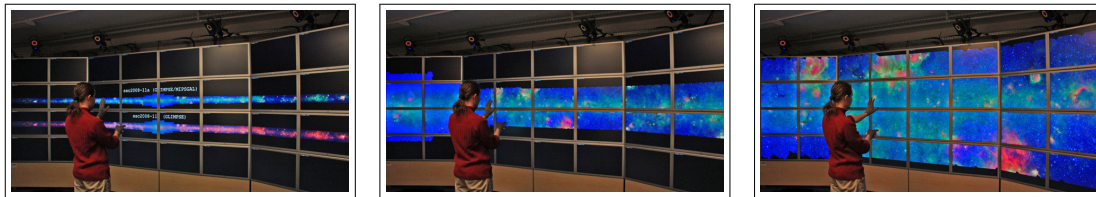


Figure 1. Panning and zooming in Spitzer's 396 032 x 12 000 = 4.7 billion pixels images of the inner part of our galaxy.

ABSTRACT

Very-high-resolution wall-sized displays offer new opportunities for interacting with large data sets. While pointing on this type of display has been studied extensively, higher-level, more complex tasks such as pan-zoom navigation have received little attention. It thus remains unclear which techniques are best suited to perform multiscale navigation in these environments. Building upon empirical data gathered from studies of pan-and-zoom on desktop computers and studies of remote pointing, we identified three key factors for the design of mid-air pan-and-zoom techniques: uni- vs. bimanual interaction, linear vs. circular movements, and level of guidance to accomplish the gestures in mid-air. After an extensive phase of iterative design and pilot testing, we ran a controlled experiment aimed at better understanding the influence of these factors on task performance. Significant effects were obtained for all three factors: bimanual interaction, linear gestures and a high level of guidance resulted in significantly improved performance. Moreover, the interaction effects among some of the dimensions suggest possible combinations for more complex, real-world tasks.

General Terms

Design, Human Factors, Performance

Author Keywords

Multi-scale interfaces, Pan & Zoom, Navigation, Wall-sized Displays, Mid-air interaction techniques.

ACM Classification Keywords

H.5.2 Information Interfaces and Presentation: User Interfaces. - Graphical user interfaces.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CHI 2011, May 7–12, 2011, Vancouver, BC, Canada.

Copyright 2011 ACM 978-1-4503-0267-8/11/05...\$10.00.

INTRODUCTION

Very-high-resolution wall-sized displays can accommodate several hundred megapixels and make it possible to visualize very large, heterogeneous datasets in many domains [1, 3, 34]. Astronomers can use them to display telescope images constructed from hundreds of thousands of frames stitched together, such as Spitzer's 4.7 billion pixels images of the inner part of our galaxy (Figure 1). Biologists can explore the docking of complex molecules. Artists can create gigapixel images, such as the 26 gigapixel panorama of Paris based on 2,346 pictures stitched together. Crisis management centers can interact with highly detailed maps of very large areas. For example, OpenStreetMap data range from a view of the world down to street level, resulting in an image that requires 18 peta ($18 \cdot 10^{15}$) pixels at its highest level of detail.

With resolutions up to 100-dpi, these LCD-based displays afford more physical forms of navigation [3, 32, 34] compared to conventional desktop setups or to lower-resolution projection-based large displays: Users simply step back to get an overview of the displayed data and walk forward to see details, including small but legible text. However, as the examples above show, datasets increase in size faster than displays increase in dimensions and pixel density. The display depicted in Figure 1 consists of thirty-two 30-inch tiled monitors and can display a “mere” 131 million pixels. NASA's Hyperwall-2, to our knowledge the largest wall built to date, only doubles that number, and does so by adding some screens that users cannot reach. Virtual navigation is thus still required, as datasets can be several orders of magnitude too large to fit on even wall-sized displays [4].

Many interaction techniques have been specifically designed to help users navigate large multiscale worlds on desktop computers, using zooming and associated interface schemes [11]. However, high-resolution wall-sized displays pose different sets of trade-offs. It is critical to their success that interaction techniques account for both the physical characteristics of the environment and the context of use, in-

CHI 2011 • Session: Mid-air Pointing & Gestures

May 7–12, 2011 • Vancouver, BC, Canada

cluding cooperative work aspects. Input should be location-independent and should require neither a hard surface such as a desk nor clumsy equipment: users should have the ability to move freely in front of the display and interact at a distance [3, 34]. This precludes use of conventional input devices such as keyboards and mice, as well as newer interaction techniques: The powerful multi-finger gestural input techniques designed by Malik *et al.* [22] were devised for interaction with lower-resolution large displays from afar. They require sitting at a desk, and are thus not optimal for displays of very high-resolution that afford more physical forms of navigation. The recent Cyclostar approach [21] is very elegant, but requires the display surface to be touch-enabled, a feature that wall-sized displays often lack. Cyclostar is also not well-suited to wall-sized displays, as it requires users to be within arm's reach of the display surface. While this is perfectly acceptable for displays up to 1.5m in diagonal such as SMART Boards™, users of larger displays such as the one in Figure 1 (5.8m in diagonal) would only see a very limited portion of the display while navigating. This lack of an overview would be a non-negligible hindrance as navigation is mostly driven by contextual information.

Our goal is to study different families of location-independent, mid-air input techniques for pan-zoom navigation on wall-sized displays. More specifically, we seek to answer questions related to the performance and subjective preferences of users, including: Beyond their almost universal appeal, do gestures performed in free space work better than those input via devices operated in mid-air? Is bimanual interaction more efficient in this context? Is it more tiring? Do circular, continuous gestures perform better than those that require clutching (restoring the hand or finger to a more comfortable posture)? We ground our work on both theoretical and experimental work on bimanual input [8, 14, 18], the influence of limb segments on input performance [2, 35], on types of gestures [25, 33] and on the integral nature, in terms of perceptual structure, of the pan-zoom task [17]. In particular, we are interested in comparing the following dimensions: bimanual vs. unimanual input; device-based vs. free-hand techniques; degrees of freedom (DOF) and associated kinesthetic and haptic feedback; and types of movements: linear gestures vs. circular, clutch-free gestures.

RELATED WORK

This work is at the intersection of many HCI research areas, including multiscale interfaces, large displays, spatial input and travel in virtual environments. This section highlights strongly related or seminal work that guided our designs and we point to relevant surveys, when available.

Large Displays

Large displays have been the focus of much research and evaluation over the last ten years. Ni *et al.* [27] survey hardware configurations, rendering techniques as well as interaction techniques for many different types of large displays.

Overall, the body of empirical work on large displays suggests that users can greatly benefit from their use. It also shows that the design of interaction techniques has to be carefully adapted to the characteristics of these displays

and to their context of use. Early studies investigated how users could benefit from larger displays in different settings. Baudisch *et al.* [4] found advantages to using a large focus+context screen over zooming and overviews to extract information from large documents such as maps and schematics of circuit boards. Improvements to spatial task performance were also identified in several complementary studies [12, 26, 31].

Other works have focused on the size and configuration of high-resolution tiled displays. Ball *et al.* [3] found that for tasks involving pan-zoom, such as navigating to a known location, searching for specific targets or looking for patterns, users perform better with larger viewport sizes that require less virtual navigation, promoting physical navigation instead. Virtual navigation was always performed with the same device: a gyroscopic mouse. Results from other recent studies suggest that large displays are also beneficial for information visualization and analysis tasks thanks to the larger amount of data that can be displayed [1, 34].

Spatial Input and Mid-air Interaction Techniques

Spatial input has been studied for years in the context of travel in immersive virtual environments and other 3D user interfaces based on virtual camera control with techniques using gloves, bimanual input and leaning, or high degrees of freedom devices [7, 24, 35]. Hinckley *et al.* [16] present a survey of design issues in spatial input, including fatigue, recalibration, clutching, motion and orientation, unimanual vs. bimanual interaction. One important issue they raise is the interdependency of all these aspects, that makes formal studies challenging, as we will see later.

Several input devices make it possible to point in mid-air on large displays: commercial devices such as gyroscopic mice, or *soap* [5], based on hardware found in a conventional optical mouse wrapped in elastic fabric. ARC-Pad [23] enables seamless absolute+relative pointing on large displays through a mobile touchscreen. The VisionWand [10] is a passive wand whose colored tips are tracked in 3D by two webcams. The multiple degrees of freedom enable a richer interaction vocabulary, that includes pan-zoom navigation.

Recent advances in motion tracking and dynamic gesture recognition technologies now make it possible to investigate freehand input techniques. Vogel and Balakrishnan [32] propose three pointing and clicking techniques that work with bare hands, with emphasis on important design characteristics such as accuracy, performance, but also comfort of use. Zigelbaum *et al.* [36] describe a gestural interface based on Oblong's *g-speak* spatial operating environment to navigate in a collection of videos arranged in a 3D interface through a set of twenty hand-centric gestures.

Multi-scale Navigation on the Desktop

Pan-zoom navigation techniques have been studied for many years in the more general context of multiscale interfaces for the desktop. Cockburn *et al.* [11] provide a thorough survey of the many zooming, overview + detail and focus + context techniques, as well as empirical work that evaluated them.

CHI 2011 • Session: Mid-air Pointing & Gestures

May 7–12, 2011 • Vancouver, BC, Canada

Of particular interest to us is the work by Guiard *et al.* on multiscale pointing. Multiscale pointing consists of panning and zooming the view so as to bring the target in view, followed by a cursor pointing action to that target [15]. They performed several empirical studies, showing that multiscale pointing obeys Fitts' law, and that performance bandwidth is proportional to view size (up to a ceiling that we far exceed on wall-sized displays). They introduced an experimental task adapted from Fitts' reciprocal pointing task, that we further adapt to take into account potential overshoots in the scale dimension. An earlier paper [6] evaluated pan-zoom performance with uni- and bimanual input, suggesting that performance is enhanced with two hands, as it affords better pan-zoom coordination. Pan-zoom navigation has however not received much attention beyond desktop interfaces, except for the recent work by Malacria *et al.* on Cyclostar [21], specifically designed for touch-enabled surfaces and discussed in more detail in the next section.

PANNING AND ZOOMING IN MID-AIR

A large body of literature is devoted to the design and evaluation of input devices that feature a high number of degrees of freedom (DOF). Available degrees of freedom have a direct impact on the potential for parallelization of actions required to achieve the task. For example, 6DOF input devices can increase the degree of parallelization of docking tasks [35], though studies report limits in terms of human capacity to handle all DOFs simultaneously. Pan and zoom is a 3DOF task: the user controls the view's position (x, y) and its scale (s). The possible solutions for mapping pan and zoom to three input channels are endless.

The film industry offers interesting and visually attractive scenarios with movies such as *Minority Report* which show users interacting via freehand gestures to navigate in a seemingly fluid and efficient way. The technology to achieve this type of interaction is now available in research laboratories and beyond [36]. However, it remains unclear how freehand gestures actually fare when compared to device-based input techniques that take advantage of the human ability to use physical tools [10] and suffer less from problems commonly associated with spatial input [16], such as precision and fatigue. Years of research in virtual reality have demonstrated that devising efficient navigation techniques for immersive virtual environments is still a challenge.

Our goal is to study families of input techniques that let users pan and zoom from any location in front of very high-resolution, wall-sized displays. We made no *a priori* assumptions about relevant metaphors or technologies and considered freehand as well as device-based techniques.

An extensive design and testing phase allowed us to limit the number of candidates for the subsequent formal evaluation. For instance, the apparently intuitive solution that consists in using two hands or two fingers to zoom with pinch and stretch gestures was considered but quickly discarded: while these gestures work well on touch-sensitive surfaces such as tablets, they are much less natural when performed in mid-air. Most importantly, they proved quite inaccurate, and

Factors	Advantages	Disadvantages
Hands	One	• One hand available for other actions
	Two	• Pan and zoom can be performed in parallel
Gesture	Linear	• Direct, natural mapping to zoom actions
	Circular	• No clutching (continuous gesture)
Degree of Guidance	1D path	• Input guided by strong haptic feedback • Mainly involves fingers
	2D surface	• Many degrees of freedom • Mainly involves fingers
	3D free hand	• Many degrees of freedom • No device

Table 1. Key Dimensions of the Design Space

tiring. Another category of techniques that was discarded are those based on first-order-of-control and operated via an elastic or isometric input device. As reported in the literature in the case of pointing, e.g., [9], our pilot tests revealed that techniques based on first-order-of-control allow for fast and comfortable coarse navigation, but perform poorly during the final precise positioning phase, causing numerous overshoots.

We eventually identified a set of twelve candidate techniques. Their design was informed by related empirical studies reported in the literature and refined through prototyping and pilot testing. These techniques can be organized according to three key dimensions forming a design space (Table 1), and introduced in the following sections. In addition to performance (task time and accuracy), we took into account other usability issues, such as fatigue and ease of use.

Unimanual vs. Bimanual Input

In their paper on the perceptual structure of multidimensional input, Jacob and Sibert claim that panning and zooming are integrally related: the user does not think of them as separate operations, but rather as a single, integral task like “focus on that area over there” [17]. Buxton and Myers [8] and later Bourgeois and Guiard [6] observed high levels of parallelism for pan-zoom operations, further supporting this argument. The level of parallelism correlates with task performance and is typically well afforded by the use of bimanual input techniques [14, 18]. While we expect bimanual techniques to outperform unimanual ones, we are still interested in comparing their performance, as the latter might still be of interest in more complex, real-world tasks that require input channels for other actions.

Linear vs. Circular Gestures

Navigating in the scale dimension (zooming in and out) is a task typically performed through vertical scroll gestures on, e.g., a mouse wheel or a touchpad. The mapping from input to command is natural, but often entails clutching as the course of mouse wheels and touchpads is very limited. An alternative consists in mapping continuous circular gestures to zooming. Clockwise gestures zoom in; counter-clockwise

CHI 2011 • Session: Mid-air Pointing & Gestures

May 7–12, 2011 • Vancouver, BC, Canada

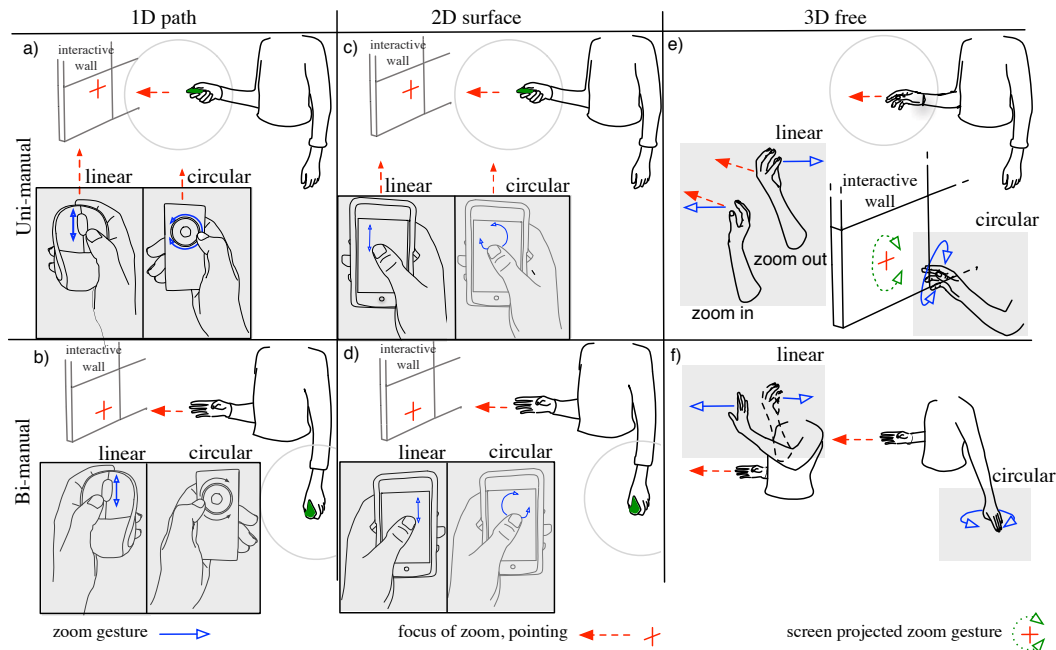


Figure 2. Matrix of the 12 techniques organized according to key characteristics: uni- vs. bimanual, degree of guidance, linear vs. circular gestures. 1D path involves guiding gestures along a particular path in space; in 2D surface gestures are made on a touch-sensitive surface; while in 3D free gestures are totally free.

gestures zoom out. Despite the less natural mapping from input to commands, such continuous, clutch-free gestures have been successfully applied to vertical scrolling in documents [25, 33], and to pan and zoom on large, touch-sensitive surfaces in CycloStar [21]. Circular gestures potentially benefit from an automatic Vernier effect [13]: as zooming is mapped to angular movements, the larger the circular gesture's radius, the greater the distance that has to be covered to make a full circle, and consequently the more precise the input.

Guidance through Passive Haptic Feedback

Two main categories of techniques have been studied for mid-air interaction on wall-sized displays: freehand techniques based on motion tracking [32, 36]; and techniques that require the user to hold an input device [5, 10, 19, 23]. Input devices provide some guidance to the user in terms of what gesture to execute, as all of them provide some sort of passive haptic feedback: A finger operating a knob or a mouse wheel follows a specific path; gestures on touch-enabled devices are made on planar surfaces. Freehand techniques, on the contrary, provide essentially no feedback to the user who can only rely on proprioception [24] to execute the gesture. We call this dimension the *degree of guidance*. Gestures can be guided to follow a particular path in space (1D path); they can be guided on a touch-sensitive surface (2D surface); or they can be totally free (3D free). These three values correspond to decreasing amounts of passive haptic feedback for the performance of input gestures.

DESIGN CHOICES

Panning. For all techniques, controlling the cursor's position is achieved naturally by ray-casting from the dominant hand to the wall display (dashed arrows in Figure 2). As mentioned earlier, first order of control was discarded for both pan and zoom operations. Panning is achieved by dragging, as in applications such as Adobe Illustrator™ or Google Maps™ with their typical hand-shaped cursor.

Zooming. As in desktop applications such as Google Maps or NASA's WorldWind, *linear* techniques zoom in by moving forward towards the display and zoom out by moving backwards; *circular* techniques zoom in by turning clockwise and zoom out by turning counter-clockwise (solid arrows in Figure 2). Pointing plays an important role when zooming, as it specifies the focus of expansion (zoom in)/contraction (zoom out). Letting users specify this focus point is very important on displays of that physical size, as they will typically not be standing right in the center. A focus of expansion implicitly located at the center of the screen would make zooming operations tedious and hard to control as every zoom operation would require multiple panning actions to compensate drifts induced by the offset focus.

Bi-manual interaction. All bimanual techniques (Figure 2, bottom row) are grounded in Guiard's study of asymmetric division of labor in bimanual actions that led to the Kinematic chain model [14]. Following the observation that motion of the dominant hand typically finds its spatial reference in the results of motion of the non-dominant-hand, we assign

CHI 2011 • Session: Mid-air Pointing & Gestures

May 7–12, 2011 • Vancouver, BC, Canada

pointing and panning to the dominant hand, while the non-dominant hand controls zoom, as is typically the case for bimanual pan-zoom techniques on the desktop [6, 8].

Input Gestures via a Device

The main limb segments involved in the input of gestures via a device are the fingers and, to a lesser extent, the forearm (for the dominant hand). This group of techniques is illustrated in Figure 2, columns *1D path* and *2D surface*.

Column *1D path* illustrates techniques that provide a high degree of guidance for executing the zooming gestures. The first row corresponds to *one handed* techniques: the device is operated by the dominant hand, which also controls pointing via ray-casting. The second row corresponds to *two handed* techniques: the dominant hand controls pointing via ray-casting, while the non-dominant hand controls zoom using the device. *linear* gestures can be input using, e.g., a wireless handheld mouse featuring a scroll wheel; *circular* gestures using, e.g., any type of handheld knob. Depressing a button on the device activates drag mode for panning.

Column *2D surface* illustrates techniques that use a touch-sensitive surface for input, providing a lesser degree of guidance. The surface is divided horizontally in two areas. Users zoom in the upper area either by moving the thumb up and down (*linear* case), or by drawing approximate circles (*circular* case). Touching the lower area activates drag mode for panning. Users just rely on proprioceptive information to switch between both areas and do not have to look at the device. These techniques can be implemented with a touch-sensitive handheld device such as a PDA or smartphone.

1D path techniques employing *circular* gestures will provide more guidance, but will not benefit from the earlier-mentioned Vernier effect, as input is constrained to one specific trajectory. However, the range of amplitudes that can be covered with the thumb is limited [30]. This should minimize the trade-off between *1D path* and *2D surface* in that respect. For *2D surface* techniques, rubbing gestures [28] were considered to avoid clutching when performing linear gestures, but were found to be impractical when performed with the thumb on a handheld touch-sensitive surface. As a technique designed specifically for thumb input, we were also interested in MicroRolls [30]. However, these were originally designed for discrete input. Cardinal MicroRolls would have had to be mapped to first order of control, which we discarded as discussed earlier, and circular MicroRolls are not precise enough for zoom control.

Input Gestures in Free Space

The main limb segments involved in performing gestures in free space are the wrist, forearm and upper arm. This group of techniques is illustrated in Figure 2, column *3D free*.

The first row illustrates *one handed* techniques using either *linear* or *circular* gestures. The technique using *circular* gestures is actually very close to the CycloStar zooming gesture, but performed in mid-air, without touching any surface. Users perform circular gestures with the dominant hand and forearm oriented toward the display. As in CycloStar, the

focus of expansion is the centroid of the round shape corresponding to the cursor's circular path, here projected on the display surface (dotted arrow in Figure 2-e). The technique using *linear* gestures consists in pushing the dominant hand forward to zoom in, as if reaching for something, with the palm towards the target. Turning the hand and pulling backward (away from the display) zooms out. Users point orthogonally to the palm of the same hand (blue arrows in Figure 2-e, left side), with the arm slightly tilted for greater comfort. The second row illustrates *two handed* techniques (Figure 2-f). The *linear* zooming gestures are similar to the ones above, but are performed with the non-dominant hand, the dominant hand still being used for pointing and specifying the focus of expansion. In the *circular* case, users adopt a potentially less tiring posture, pointing at the floor with their non-dominant hand and making circular movements. All other postures and movements being ignored by the system for the non-dominant hand, the user can easily clutch. Several options can be considered for engaging drag mode: specific hand postures such as pinching, or using a small wireless actuator (e.g., a button).

EXPERIMENT

We conducted an experiment using a $[2 \times 2 \times 3]$ within-subjects design with three primary factors: $HANDEDNESS \in \{OneHanded, TwoHanded\}$, $GESTURE \in \{Circular, Linear\}$, and $GUIDANCE \in \{1DPath, 2DSurface, 3DFree\}$ to evaluate the 12 unique interaction techniques described above. We controlled for potential distance effects by introducing the $DISTANCE$ between two consecutive targets as a secondary within-subjects factor. We systematically varied these factors in the context of a multiscale navigation task within a wall-sized display environment.

Measures include performance time and number of overshoots, treated as errors. Overshoots occur when participants zooms beyond the target zoom level, and indicate situations in which the participant has less precision of control over the level of zoom. For instance, from an overview of Canada, zooming down to street level in Google Maps when what the user actually wanted was to get an overview of Vancouver.

Hypotheses

Based on the research literature and our own experience with the above techniques, we made the following 7 hypotheses.

Handedness: prior work [6, 8, 15, 18] suggests that two-handed gestures will be faster than one-handed gestures (*H1*) because panning and zooming are complementary actions, integrated into a single task [17]. Two-handed gestures should also be more accurate and easier to use (*H2*).

Gesture: Linear gestures should map better to the zooming component of the task, but should eventually be slower because of clutching, the limited action space compared to zoom range requiring participants to repeatedly reposition their hand/finger (*H3*). Prior work [25, 33] suggests that users will prefer clutch-free circular gestures (*H4*).

Device vs. Free Space: Zhai et al. [35] suggest that techniques using the smaller muscle groups of fingers should be

CHI 2011 • Session: Mid-air Pointing & Gestures

May 7–12, 2011 • Vancouver, BC, Canada

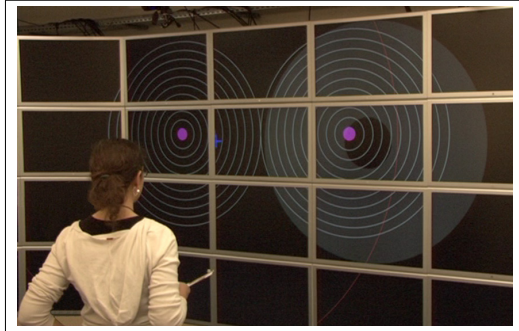


Figure 3. Participant performing the task

more efficient than those using upper limb segments. Balakrishnan *et al.* [2] moderate this observation with findings suggesting that the fingers are not performing better than forearm or wrist for a reciprocal pointing task. Nevertheless, they acknowledge that differences exist in the motor system's ability to control the different limb segments. Based on the gestures to be performed and taking into account the physical size and mass of the segments involved, we predicted that techniques using fingers (*1DPath* and *2DSurface* conditions), should be faster than those requiring larger muscle groups (hands and arms, *3DFree* conditions) (*H5*).

We also predicted that *1DPath* gestures would be faster, with fewer overshoots than techniques with lesser haptic feedback, i.e., *2DSurface* and *3DFree* (*H6*). Finally, we predicted that *3DFree* gestures would be more tiring (*H7*).

Participants

We recruited 12 participants (1 female), ranging in age from 20 to 30 years old (average 24.75, median = 25). All are right-handed daily computer users. None are color-blind.

Apparatus

Hardware. The display wall (Fig. 1 and 3) consists of 32 high-resolution 30" LCDs laid out in an 8×4 matrix, 5.5 meters wide and 1.8 meters high. It can display 20480×6400 pixels. A cluster of 16 computers, each with two high-end nVidia 8800GT graphics cards, communicate via a dedicated high-speed network through a front-end computer. Our goal is to identify the performance characteristics of each technique from the user's perspective. It is thus essential that each technique operates equally well from a purely technological perspective. We use a VICON motion-capture system to track passive IR retroreflective markers and provide 3D object coordinates with sub-millimeter accuracy at 200Hz (although gesture recognition technologies are constantly improving, such a system is still necessary to get reliable and precise 3D position/orientation information). The *Linear 1DPath* condition uses the wheel of a wireless Logitech M305 mouse (Fig. 2-a,b). The *Circular 1DPath* condition uses a wireless Samsung SM30P pointing device, normally used for presentations (Fig. 2-a,b). All *2DSurface* conditions use an iPod Touch. So as to avoid failures from gesture segmentation algorithms that would impact task performance in an uncontrolled manner, we use an explicit mode switch to unambiguously engage drag mode (panning).

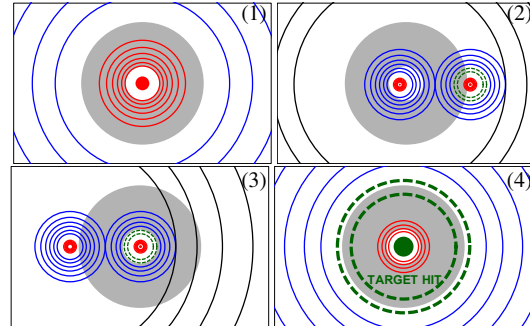


Figure 4. Task (schematic representation using altered colors): (1) Groups of concentric circles represent a given position and zoom level. (2) Zooming out until the neighboring set of circles appears. (3-4) Pan and zoom until the target (green inner disc and circles, dashed for illustration purposes only) is positioned correctly with respect to the stationary gray ring.

As mentioned earlier, we use the device's main button for *1DPath* conditions, and the lower area of the touch-sensitive surface for *2DSurface* conditions. While in real-world applications we would use specific hand postures such as pinching in *3DFree* conditions, for the sake of robustness we use a wireless mouse button whose activation is seamlessly integrated with the gesture.

Software. The experiment was written in Java 1.5 running on Mac OS X and was implemented with the open source ZVTM toolkit [29] (<http://zvtm.sf.net>) modified to run on clusters of computers driving display walls. Touchstone [20] was used to manage the experiment.

Pan-Zoom Task

The task is a variation of Guiard *et al.*'s multiscale pointing task [15], adapted to take overshoots into account. Participants navigate through an abstract information space made of two groups of concentric circles: the *start group* and the *target group*. Each group consists of seven series of 10 concentric circles symbolizing different zoom levels, each designated by a different color (Fig. 4.2). The target group features two additional green circles (dashed in Fig. 4.4) and a disc, referred to as C_1 , C_2 and C_3 from smallest to largest.

Participants start at a high zoom level in the start group (Fig. 4.1). They zoom out until the neighboring target group appears (Fig. 4.2). It may appear either on the left or right side of the start group. Then they pan and zoom into the target group until they reach the correct zoom level and the target is correctly centered. A stationary gray ring symbolizes the correct zoom level and position (Fig. 4-(1-4)). Its radii are $r_1 = 4400$ and $r_2 = 12480$ pixels. All three criteria must be met for the trial to end: *A*) C_1 is fully contained within the stationary ring's hole (radius = r_1), *B*) $radius(C_2) < r_2$, *C*) $radius(C_3) > r_2$. Overshoots occur when the zoom level is higher than the maximum level required to meet criteria *B* and *C*, in which case participants have to zoom out again (C_1 becomes white instead of green in that situation). When all conditions are met, the message TARGET HIT appears and the thickness of C_2 and C_3 is increased (Fig. 4.4). The trial

CHI 2011 • Session: Mid-air Pointing & Gestures

May 7–12, 2011 • Vancouver, BC, Canada

ends when the position and zoom level have stabilized for at least 1.2 seconds (all trials must be successfully completed).

Procedure

The experiment presents each subject with six replications of each of the 12 techniques at three DISTANCES. The experiment is organized into four sessions that each present three techniques: One combination of the GESTURE and HANDEDNESS factors and all three degrees of GUIDANCE. Each session lasts between 30 and 90 minutes, depending on techniques and participant. Participants are required to wait at least one hour between two consecutive sessions, and to complete the whole experiment within four days or fewer, with a maximum of two sessions per day to avoid too much fatigue and boredom. Participants stand 1.7m from the wall and are asked to find a comfortable position so they can perform gestures quickly, but in a relaxed way.

Practice Condition: Participants are given a brief introduction at the beginning of the first session. Each technique begins with a practice condition, with trials at three different DISTANCES: (49 920, 798 720 and 12 779 520 pixels). Measures for the experimental condition start as soon as 1) participants feel comfortable and 2) task performance time variation for the last four trials is less than 30% of the task time average in that window.

Experimental Condition: Each technique is presented in a block of 18 trials consisting of 6 replications at each DISTANCE. Trials, blocks and sessions are fully counter-balanced within and across subjects, using a Latin square design.

Measures: We measure movement time *MT* and number of overshoots for each of 2592 trials: 2 GESTURE $\times$ 2 HANDEDNESS $\times$ 3 GUIDANCE $\times$ 3 DISTANCE $\times$ 12 participants $\times$ 6 replications. Participants also answer questions, based on a 5-point Likert scale, about their perceived *performance*, *accuracy*, *ease of learning*, *ease of use*, and *fatigue*. They rank the techniques with respect to the GUIDANCE factor after each session. When they have been exposed to both conditions of HANDEDNESS or GESTURE, they rank those as well. After the last session, they rank the techniques individually and by factor. Participants are encouraged to make additional observations and comments about any of the above.

Results and Discussion: Movement Time

Prior to our analysis, we checked the performance for unwanted effects from secondary factors. We checked for individual performance differences across subjects and found that, for all 12 participants, movement time and number of overshoots were perfectly correlated with the overall performance measures. As expected, movement time data are skewed positively; replications of unique experimental conditions are thus handled by taking the median (note that taking the mean yields similar results). In all remaining analysis, we handled participant as a random variable, using the standard repeated measures REML technique. We found no significant fatigue effect although we did find a significant learning effect across sessions. Participants performed about 1.4 s more slowly in the first session and then became slightly faster over the next three sessions. However, we found no significant interaction between session orders and

Factor	DF, DFDen	F Ratio	p
HANDS	1,11	24.65	0.0004 *
GESTURE	1,11	42.87	< 0.0001 *
GUIDANCE	2,22	58.80	< 0.0001 *
DIST	2,22	228.8	< 0.0001 *
HANDS $\times$ GESTURE	1,11	2.060	0.1790
HANDS $\times$ GUIDANCE	2,22	4.914	0.0172 *
GESTURE $\times$ GUIDANCE	2,22	10.38	0.0007 *
GESTURE $\times$ DIST	2,22	17.27	< 0.0001 *
HANDS $\times$ DIST	2,22	11.57	0.0004 *
GUIDANCE $\times$ DIST	4,44	3.828	0.0094 *
HANDS $\times$ GESTURE $\times$ GUIDANCE	2,22	1.127	0.3420
HANDS $\times$ GESTURE $\times$ DIST	2,22	0.790	0.4661
HANDS $\times$ GUIDANCE $\times$ DIST	4,44	0.650	0.6301
GESTURE $\times$ GUIDANCE $\times$ DIST	4,44	3.750	0.0104 *
HANDS $\times$ GESTURE $\times$ GUIDANCE $\times$ DIST	4,44	1.049	0.3929

Table 2. Results of the full factorial ANOVA for *MT*.

main factors. As the factors were counter-balanced, this created no adverse effects in the analysis.

Table 2 details results of the full factorial ANOVA for the model $MT \sim HANDS \times GUIDANCE \times GESTURE \times DIST \times Rand(\text{Participant})$. We observe that HANDS has a significant effect on *MT* (Figure 5-a¹). A post-hoc Tukey test shows that *TwoHanded* gestures are significantly faster than *OneHanded* gestures (avg. 9690 ms vs. 11869 ms). We found a significant interaction effect of HANDS $\times$ GUIDANCE (Figure 5-a). The interaction does not change the significance of the post-hoc test, but indicates that the magnitude of the difference is greater for *3DFree* than for *2DSurface* and greater for *2DSurface* than for *1DPath* techniques.

Unsurprisingly, performance data strongly support (*H1*): all other conditions being equal, two-handed techniques are consistently faster than one-handed techniques. An interesting observation is that using two hands is more advantageous when the degree of guidance for achieving gestures is low.

GUIDANCE has a significant effect on *MT* (Figure 5-b). A post-hoc Tukey test shows that *1DPath* (avg. 9511 ms) is significantly faster than *2DSurface* (10894 ms), which in turn is significantly faster than *3DFree* (11934 ms). This time the HANDS $\times$ GUIDANCE interaction changes the significance of the test (Figure 5-b). The difference is that a post-hoc Tukey test shows no significant difference between *2DSurface* and *3DFree* for *TwoHanded*.

Both hypotheses (*H5*) and (*H6*) are supported: involving smaller muscle groups improves performance; providing higher guidance further contributes to this. However, this effect is less pronounced in *TwoHanded* conditions. This confirms the previous observation that a higher degree of guidance is especially useful when a single hand is involved.

GESTURE also has a significant effect on *MT*. A post-hoc Tukey test shows that *Linear* movements (avg. 9384 ms) performed significantly faster than *Circular* gestures (12175 ms). However, we have a strong significant interaction of GESTURE $\times$

¹Error bars in all the figures represent the 95% confidence limit of the mean of the medians per participants ($\pm StdErr \times 1.96$).

CHI 2011 • Session: Mid-air Pointing & Gestures

May 7–12, 2011 • Vancouver, BC, Canada

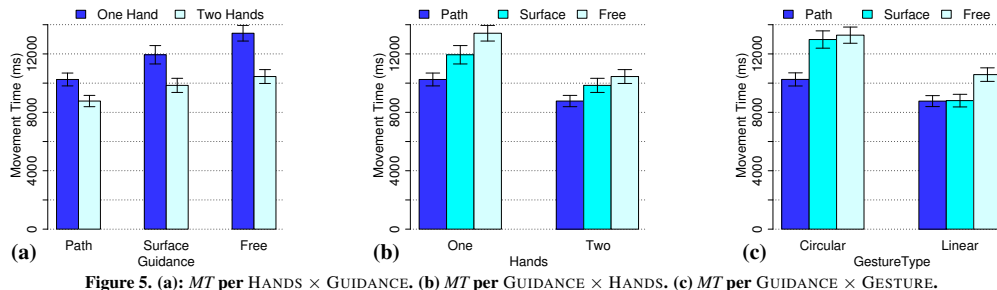


Figure 5. (a): MT per HANDS $\times$ GUIDANCE. (b) MT per GUIDANCE $\times$ HANDS. (c) MT per GUIDANCE $\times$ GESTURE.

GUIDANCE (Figure 5-c). A post-hoc Tukey test shows that (i) for *Circular* gestures: *1DPath* guidance is faster than both *2DSurface* and *3DFree* with no significant difference between *2DSurface* and *3DFree*; (ii) for *Linear* gestures, there is no significant difference between *1DPath* and *2DSurface*, but a significant difference between *2DSurface* and *3DFree*; (iii) for *1DPath* guidance there is no significant difference between *Circular* and *Linear* gestures, but there is a significant difference between *Circular* and *Linear* for *2DSurface* and *3DFree* guidance.

Surprisingly, *Linear* gestures are generally faster than *Circular* ones. (*H3*), that claimed that *Linear* gestures should be slower because of clutching, is not supported. Performance differences between gesture types are however affected by the degree of guidance: *Circular* gestures with *1DPath* guidance (e.g., a knob) are comparable to *Linear* gestures with low guidance. We tentatively explain the lower performance of *Circular* gestures with *2DSurface* guidance by the difficulty of performing circular gestures with the thumb [30], also observed here.

Another interesting observation is that our analogue of CycloStar in mid-air (*Circular* gestures with *3DFree* guidance) performs poorly. It seems that the lack of a surface to guide the gesture significantly degrades this technique's usability. Another factor contributing to its poor performance in our study is likely related to overshoots, as discussed below.

As expected, distance to target (DIST) has a significant effect on MT. A post-hoc Tukey test shows that MT increases significantly with distance. There are several significant interactions between DIST and the main factors (Fig. 6), but none of these change the relative performance ordering for the main factors. These interactions are due to a change in the magnitude of the difference across conditions, confirming that the choice of an efficient technique is of increasing importance as the task becomes harder.

Results and Discussion: Overshoots

As detailed earlier in the description of task design, overshoots correspond to zooming beyond the target zoom level and are treated as errors. We consider the model $\text{Overshoots} \sim \text{HANDS} \times \text{GUIDANCE} \times \text{GESTURE} \times \text{DIST} \times \text{Rand}(\text{Participant})$.

We observe significant simple effects on *Overshoots* for GESTURE ($F_{1,11} = 21.04, p = 0.0008$) and GUIDANCE ($F_{2,22} = 53.80, p < 0.0001$), and one significant interaction effect for GESTURE $\times$ GUIDANCE ($F_{2,22} = 8.63, p = 0.0017$). *Circular* gestures exhibit more overshoots than *Linear* gestures (1.65 vs. 2.71).

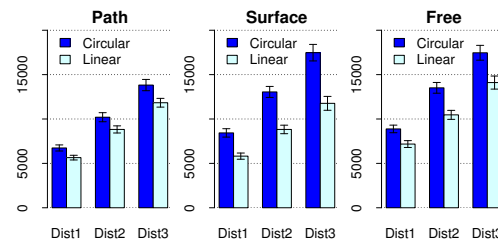


Figure 6. MT per DIST $\times$ GESTURE, for each GUIDANCE

2DSurface gestures exhibit more overshoots than *1DPath* and *3DFree* gestures (3.75 for *2DSurface* vs. 1.52 for *1DPath*, and 1.26 for *3DFree*). There is a significant difference between *Linear* and *Circular* gestures for *2DSurface* and *3DFree*, but not *1DPath*. Moreover, overshoots exhibit the same interaction effect for *2DSurface* gestures: *Circular 2DSurface* result in significantly more overshoots than *Linear 2DSurface* (4.68 vs. 2.82).

The observed higher number of overshoots for *Circular* techniques helps explain the generally lower MT performance measured for this type of gestures. The best-fitting ellipse algorithm involved in the recognition of *Circular* gestures has an inherently higher cost of recovery, introducing a delay when reversing course. The poor performance of our analogue of CycloStar is at least partially due to this, knowing that there was a major difference between the zooming experiment reported in [21] and the present one: we included overshoots in our task design, whereas the CycloStar experiment apparently did not (there is no report of such a measure in task design or results analysis), thus ignoring this issue.

Results and Discussion: Qualitative Results

Qualitative data confirms our results. Participants generally preferred *TwoHanded* to *OneHanded* techniques (8/12) and *Linear* to *Circular* gestures (10/12). Subjective preferences about degree of guidance were mixed, with 4 participants preferring the high degree of guidance provided by *1DPath* techniques, only 1 for both of *2DSurface* and *3DFree* techniques, and all others expressing no particular preferences. Looking at the details of answers to our 5-point Likert scale questions about perceived speed, accuracy, ease of use and fatigue, significant results ($p < 0.002$) were obtained only for degree of GUIDANCE, with *1DPath* being consistently rated higher than *2DSurface* and *3DFree*; and for HANDS, *TwoHanded* techniques being considered less tiring than *OneHanded* techniques ($p < 0.03$).

CHI 2011 • Session: Mid-air Pointing & Gestures

May 7–12, 2011 • Vancouver, BC, Canada

GROUP	HANDS	GESTURE	GUIDANCE	Figure	MT (ms)
Gr1	TwoHanded	Linear	2DSurface	2-d	8 100
	TwoHanded	Linear	1DPath	2-b	8 377
Gr2	OneHanded	Linear	1DPath	2-a	9 160
	TwoHanded	Circular	1DPath	2-b	9 168
	TwoHanded	Linear	3DFree	2-f	9 185
	OneHanded	Linear	2DSurface	2-c	9 504
Gr3	OneHanded	Circular	1DPath	2-a	11 340
	TwoHanded	Circular	2DSurface	2-d	11 591
	TwoHanded	Circular	3DFree	2-f	11 718
	OneHanded	Linear	3DFree	2-e	11 981
Gr4	OneHanded	Circular	2DSurface	2-c	14 380
	OneHanded	Circular	3DFree	2-e	14 851

Table 3. Groups of techniques according to MT

Comments from participants suggest that in the *OneHanded* condition, zoom gestures interfere with pointing as they introduce additional hand jitter and consequently lower accuracy. Some participants also commented that pointing and zooming were confounded in the *OneHanded* conditions, making the techniques difficult to use (*H2*). However, two participants strongly preferred one-handed gestures, arguing that they were less complex and less tiring. They assumed their performance was better (even though it was not), probably because they experienced more overshoots in the *two handed* condition, which may have led to their conclusions. One of them mentioned that for the *one handed* condition there was “no need for coordination”; techniques were “more relaxed” and made it “easier to pan and zoom”.

Linear gestures were preferred to *Circular* ones, participants commenting that circular gestures were difficult to perform without guidance, that circular gestures for zooming interfered with linear gestures for panning, and that circular gestures were hard to map to zoom factor. All but one participants preferred linear gestures overall although one commented that he liked “the continuity of circular gestures”. Others commented that “making good circles without a guide is hard” and did not like having to turn their hands. These findings contradict our hypothesis that users would prefer clutch-free circular gestures (*H4*). This hypothesis was based on observations made for techniques operated on a desktop, not in mid-air, and involved different limb segments. In many of our conditions, the gestures had to be performed with the thumb, and were thus more complex to achieve than when using, e.g., the index finger in conjunction with hand or forearm movements. Several participants commented on this interaction effect: “*[It is] too hard to do circle gestures without a guide*”, “*Linear movements are easier on the iPod*” and “*[Is it] impossible to do circular movements on a surface, maybe with some oil?*”.

Finally, as hypothesized (*H7*), participants found *1DPath* guidance least tiring while *3DFree* caused the most fatigue.

Results and Discussion: Individual Techniques

The analysis of variance for the model $MT \sim \text{HANDS} \times \text{GUIDANCE} \times \text{GESTURE} \times \text{DIST} \times \text{Rand}(\text{Participant})$ does not show a significant triple interaction between the three main factors (Table 2). Formally, we cannot say more than the above about the ranking of the twelve techniques. However, based on the results about *MT* above, we can observe four distinct

groups of techniques, shown in Table 3. As a side note, if we consider the model $MT \sim \text{GROUP} \times \text{Rand}(\text{Participant})$, the ANOVA shows a significant effect of *GROUP* ($F_{3,33} = 65.35$, $p < 0.0001$) and a post-hoc Tukey test shows a significant difference between each groups.

Gr1 contains the two fastest techniques with similar *MT*: *TwoHanded*, *Linear* gestures with either *2DSurface* or *1DPath* degrees of guidance. Optimal performance in terms of movement time implies the use of two hands and a device to guide gestural input.

Gr2 contains the four techniques that come next and also have close *MT*: the *OneHanded* version of the two fastest techniques, the *TwoHanded Circular 1DPath* and the *TwoHanded Linear 3DFree* techniques. Techniques in this group are of interest as they exhibit a relatively good level of performance while broadening possible choices for interaction designers. For instance, the unimanual techniques in this group make one hand available to perform other actions. The *3DFree* technique is also of interest as it does not require the user to hold any equipment and is generally appealing to users.

Gr3 contains techniques that again have very close *MT* but about 2.3 s slower than the techniques of *Gr2*. This group consists of *OneHanded Circular 1DPath*, *TwoHanded Circular 2DSurface* and *3DFree*, and *OneHanded Linear 3DFree*. Techniques in this group are of lesser interest, except maybe for the *OneHanded Linear 3DFree* technique, which is the fastest unimanual technique using gestures performed in free space.

Gr4 contains the 2 techniques performing worst, *OneHanded Circular 2DSurface* and *3DFree*. These are significantly slower than all others, about 3 s slower than the techniques of *Gr3* and about 6 s slower than the techniques of *Gr1*. Our data suggest that these techniques should be rejected.

SUMMARY AND FUTURE WORK

We studied different families of location-independent, mid-air input techniques for pan-zoom navigation on wall-sized displays. After an extensive exploratory design phase, we identified the following key factors for the design of such techniques: handedness (uni- vs. bimanual input), gesture type (linear or circular), and level of guidance (movements restricted to a 1D path, a 2D surface or free movements in 3D space). We systematically evaluated each combination of these factors through a controlled experiment in which participants performed pan-and-zoom navigation in an abstract, very large multiscale environment, with distances up to 12 million pixels.

Experimental results identify several successful mid-air input techniques that can be used to navigate efficiently in very large datasets on wall-sized displays. In addition to identifying groups of alternative techniques based on performance, but each with specific characteristics, the experiment also suggests clear results with respect to the factors that constitute our design space. For example, despite their inherent and almost universal appeal, gestures performed in free space prove to be generally less efficient and more prone to

CHI 2011 • Session: Mid-air Pointing & Gestures

May 7–12, 2011 • Vancouver, BC, Canada

fatigue than device-based input techniques. Adding guidance to input gestures increases, rather than decreases, accuracy. In accordance with the research literature, bimanual input techniques perform very well. Unimanual techniques perform honorably, and may still be considered in contexts of use where, for example, tools must be held in one hand to perform a domain/task specific action. A more surprising result is the generally higher efficiency of linear gestures when compared to circular, clutch-free gestures.

As future work, we plan to investigate how these pan-zoom techniques combine with other interaction techniques. Indeed, in real-world applications, users must also handle text entry, menu selection, copy and paste, drag and drop, and other activities. This implies trade-offs among techniques: a technique with optimal performance in this experiment may prove less easy to integrate with other techniques because of its requirements in terms of handedness or type of device. We have started to explore these questions in the context of real-world activities involving scientists visualizing and manipulating extremely large sets of multi-scale data.

ACKNOWLEDGEMENTS

We wish to thank Clément Pillias and Romain Primet, who helped implement the experiment on the wall-sized display, as well as Caroline Appert, Stéphane Huot and the anonymous reviewers for their feedback on drafts of this paper. We also wish to thank all the volunteers who participated in our experiments. This research was supported by a Région Île-de-France/Digiteo grant.

REFERENCES

1. C. Andrews, A. Endert, and C. North. Space to think: large high-resolution displays for sensemaking. In *Proc. CHI '10*, 55–64. ACM, 2010.
2. R. Balakrishnan and I. S. MacKenzie. Performance differences in the fingers, wrist, and forearm in computer input control. In *Proc. CHI '97*, 303–310. ACM, 1997.
3. R. Ball, C. North, and D. Bowman. Move to improve: promoting physical navigation to increase user performance with large displays. In *Proc. CHI '07*, 191–200. ACM, 2007.
4. P. Baudisch, N. Good, V. Bellotti, and P. Schraedley. Keeping things in context: a comparative evaluation of focus plus context screens, overviews, and zooming. In *Proc. CHI '02*, 259–266. ACM, 2002.
5. P. Baudisch, M. Sinclair, and A. Wilson. Soap: a pointing device that works in mid-air. In *Proc. UIST '06*, 43–46. ACM, 2006.
6. F. Bourgeois and Y. Guiard. Multiscale pointing: facilitating pan-zoom coordination. In *CHI '02 EA*, 758–759. ACM, 2002.
7. D. A. Bowman, D. Koller, and L. Hodges. Travel in immersive virtual environments: An evaluation of viewpoint motion control techniques. In *Proc. VRAIS '97*, 45–52. IEEE, 1997.
8. W. Buxton and B. Myers. A study in two-handed input. *SIGCHI Bull.*, 17(4):321–326, 1986.
9. B. Campbell, K. O'Brien, M. Byrne, and B. J. Bachman. Fitts' law predictions with an alternative pointing device (wiimote®). *Proc. Hum. Factors Ergon. Soc. Meeting*, 52(19):1321–1325, 2008.
10. X. Cao and R. Balakrishnan. Visionwand: interaction techniques for large displays using a passive wand tracked in 3d. In *Proc. UIST '03*, 173–182. ACM, 2003.
11. A. Cockburn, A. Karlson, and B. B. Bederson. A review of overview+detail, zooming, and focus+context interfaces. *ACM CSUR*, 41(1):1–31, 2008.
12. M. Czerwinski, D. Tan, and G. Robertson. Women take a wider view. In *Proc. CHI '02*, 195–202. ACM, 2002.
13. K. Evans, P. Tanner, and M. Wein. Tablet-based valuator that provide one, two, or three degrees of freedom. In *Proc. SIGGRAPH '81*, 91–97. ACM, 1981.
14. Y. Guiard. Asymmetric division of labor in human skilled bimanual action: The kinematic chain as a model. *J. Motor Behav.*, 19:486–517, 1987.
15. Y. Guiard and M. Beaudouin-Lafon. Target acquisition in multiscale electronic worlds. *IJHCS*, 61(6):875–905, 2004.
16. K. Hinckley, R. Pausch, J. Goble, and N. Kassell. A survey of design issues in spatial input. In *Proc. UIST '94*, 213–222. ACM, 1994.
17. R. Jacob and L. Sibert. The perceptual structure of multidimensional input device selection. In *Proc. CHI '92*, 211–218. ACM, 1992.
18. A. Leganchuk, S. Zhai, and W. Buxton. Manual and cognitive benefits of two-handed input: an experimental study. *ACM ToCHI*, 5(4):326–359, 1998.
19. R. Lindeman, J. Sibert, and J. Hahn. Hand-held windows: Towards effective 2d interaction in immersive virtual environments. In *Proc. VR '99*, 205–212. IEEE, 1999.
20. W. E. Mackay, C. Appert, M. Beaudouin-Lafon, O. Chapuis, Y. Du, J.-D. Fekete, and Y. Guiard. Touchstone: exploratory design of experiments. In *Proc. CHI '07*, 1425–1434. ACM, 2007.
21. S. Malacria, E. Lecolinet, and Y. Guiard. Clutch-free panning and integrated pan-zoom control on touch-sensitive surfaces: the cyclostar approach. In *Proc. CHI '10*, 2615–2624. ACM, 2010.
22. S. Malik, A. Ranjan, and R. Balakrishnan. Interacting with large displays from a distance with vision-tracked multi-finger gestural input. In *Proc. UIST '05*, 43–52. ACM, 2005.
23. D. McCallum and P. Irani. Arc-pad: absolute-relative cursor positioning for large displays with a mobile touchscreen. In *Proc. UIST '09*, 153–156. ACM, 2009.
24. M. Mine, F. Brooks, Jr., and C. Sequin. Moving objects in space: exploiting proprioception in virtual-environment interaction. In *Proc. SIGGRAPH '97*, 19–26. ACM Press/Addison-Wesley, 1997.
25. T. Moscovich and J. F. Hughes. Navigating documents with the virtual scroll ring. In *Proc. UIST '04*, 57–60. ACM, 2004.
26. T. Ni, D. Bowman, and J. Chen. Increased display size and resolution improve task performance in information-rich virtual environments. In *Proc. GI '06*, 139–146. Canadian Inf. Proc. Soc., 2006.
27. T. Ni, G. Schmidt, O. Staadt, M. Livingston, R. Ball, and R. May. A survey of large high-resolution display technologies, techniques, and applications. In *Proc. VR '06*, 223–236. IEEE, 2006.
28. A. Olwal, S. Feiner, and S. Heyman. Rubbing and tapping for precise and rapid selection on touch-screen displays. In *Proc. CHI '08*, 295–304. ACM, 2008.
29. E. Pietriga. A toolkit for addressing hci issues in visual language environments. In *Proc. VLHCC '05*, 145–152. IEEE, 2005.
30. A. Roudaut, E. Lecolinet, and Y. Guiard. Microrolls: expanding touch-screen input vocabulary by distinguishing rolls vs. slides of the thumb. In *Proc. CHI '09*, 927–936. ACM, 2009.
31. D. Tan, D. Gergle, P. Scupelli, and R. Pausch. Physically large displays improve performance on spatial tasks. *ACM ToCHI*, 13(1):71–99, 2006.
32. D. Vogel and R. Balakrishnan. Distant freehand pointing and clicking on very large, high resolution displays. In *Proc. UIST '05*, 33–42. ACM, 2005.
33. E. Wherry. Scroll ring performance evaluation. In *CHI '03 EA*, 758–759. ACM, 2003.
34. B. Yost, Y. Hacıahmetoglu, and C. North. Beyond visual acuity: the perceptual scalability of information visualizations for large displays. In *Proc. CHI '07*, 101–110. ACM, 2007.
35. S. Zhai, P. Milgram, and W. Buxton. The influence of muscle groups on performance of multiple degree-of-freedom input. In *Proc. CHI '96*, 308–315. ACM, 1996.
36. J. Zigelbaum, A. Browning, D. Leithinger, O. Bau, and H. Ishii. g-stalt: a chirocentric, spatiotemporal, and telekinetic gestural interface. In *Proc. TEI '10*, 261–264. ACM, 2010.

Rapid Development of User Interfaces on Cluster-Driven Wall Displays with jBricks

Emmanuel Pietriga^{1,2} Stéphane Huot^{2,1} Mathieu Nancel^{2,1} Romain Primet¹

¹INRIA
F-91405 Orsay, France

²LRI - Univ Paris-Sud & CNRS
F-91405 Orsay, France

ABSTRACT

Research on cluster-driven wall displays has mostly focused on techniques for parallel rendering of complex 3D models. There has been comparatively little research effort dedicated to other types of graphics and to the software engineering issues that arise when prototyping novel interaction techniques or developing full-featured applications for such displays. We present jBricks, a Java toolkit that integrates a high-quality 2D graphics rendering engine and a versatile input configuration module into a coherent framework, enabling the exploratory prototyping of interaction techniques and rapid development of post-WIMP applications running on cluster-driven interactive visualization platforms.

General Terms

Design, Human Factors, Performance

Keywords

Wall Displays, Clusters, Interaction, Toolkit, Prototyping

ACM Classification Keywords

H.5.2 Information Interfaces and Presentation: User Interfaces. - Graphical user interfaces.

INTRODUCTION

Over the last decade, wall-sized displays have evolved from experimental, CRT monitor-based setups to sophisticated arrays of tiled projectors or LCD panels. The latter are often called *ultra-high-resolution* displays to emphasize their significantly higher display capacity compared to projector-based *very-high-resolution* displays. They typically accommodate several hundred megapixels, and are driven by clusters of computers. As an example, the setup depicted in Figure 1 uses 32+1 graphic processing units in 16+1 computers to display $20480 \times 6400 \simeq 131$ megapixels on a $5.5m \times 1.8m$ surface ($\simeq 100dpi$). These displays enable the visualization of truly massive datasets. They can represent the data with a high level of detail while retaining context [14], and enable the juxtaposition of data in various forms. To make them interactive, wall-sized displays are increasingly coupled with advanced input devices, e.g., motion-tracking sys-

tems, wireless multitouch devices, in order to enable multi-device and/or multi-user interaction with the displayed data [14, 15]. These interactive ultra-high-resolution displays can be used in many application domains, including command and control centers, geospatial imagery, scientific visualization, collaborative design and public information displays.

These new environments pose new research challenges. From a *computer graphics perspective*: how to render complex graphics at high frame rates, taking advantage of the cluster's computing and rendering power. From a *human-computer interaction perspective*: how to design effective visualizations that take advantage of the specific characteristics of large, ultra-high-resolution surfaces; how to design interaction techniques that are well-adapted to this particular context of use, and how to handle the multiple and heterogeneous input devices and modalities typically used in this context. Finally, from a *software engineering perspective*: how to enable the rapid prototyping, development, testing and debugging of interactive applications running on clusters of computers, providing the right abstractions.

In this paper, we focus on the latter research question, that we consider essential to foster more research and development from the HCI perspective. We present jBricks, a Java toolkit for the development of post-WIMP applications executed on cluster-driven wall displays, that extends and integrates a high-quality 2D graphics rendering engine and a versatile input management module into a coherent framework hiding low-level details from the developer. The goal of this framework is to ease the development, testing and debugging of interactive visualization applications. It also offers an environment for the rapid prototyping of novel interaction techniques and their evaluation through controlled experiments, such as the one we recently conducted about mid-air pan-and-zoom techniques for wall-sized displays [14].

Background and Motivation

The parallel-rendering techniques developed over the last ten years enable the efficient display of 3D graphics on tiled displays driven by clusters of computers. This is usually done by sending already rendered images to the cluster nodes, or by sending geometry and performing compositing operations to produce the final wall-sized image. Different techniques exist, including *sort-first* and *sort-last* pipelines as well as various hybrid solutions. Well-known frameworks include Chromium [9], Equalizer [8] and SAGE [11]. See Ni *et al.* [15] for a comprehensive survey.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

EICS'11, June 13–16, 2011, Pisa, Italy.

Copyright 2011 ACM 978-1-4503-0670-6/11/06...\$10.00.

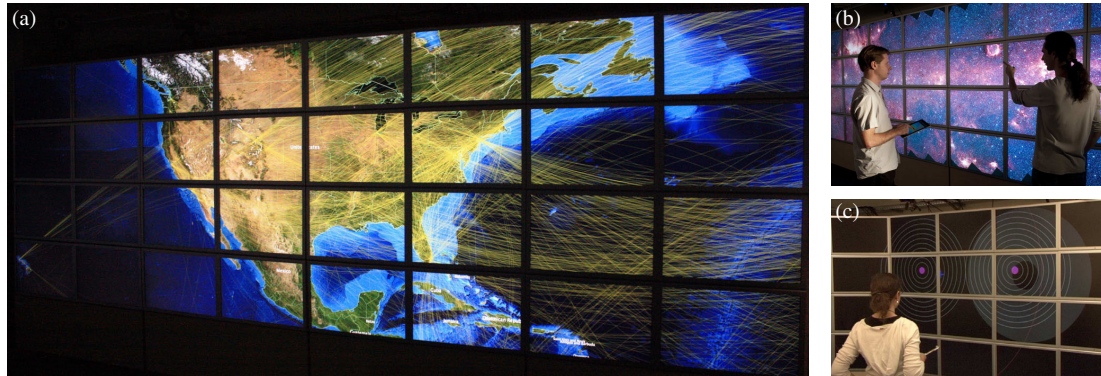


Figure 1. jBricks applications running on the WILD platform (32 tiles for a total resolution of $20\,480 \times 6\,400$ pixels). (a) Zoomed-in visualization of the North-American part of the world-wide air traffic network (1 200 airports, 5 700 connections) overlaid on NASA's Blue Marble Next Generation images ($86\,400 \times 43\,200$ pixels) augmented with country borders ESRI shapefiles. (b) Panning and zooming in Spitzer's Infrared Milky Way ($396\,032 \times 12\,000$ pixels). (c) Controlled laboratory experiment for the evaluation of mid-air multi-scale navigation techniques [17].

However, not all wall display applications use 3D graphics. With the introduction of ultra-high resolution, high-quality 2D graphics open wall-sized displays to new applications, e.g., in astronomy, geospatial intelligence and visual analytics at large, to give a few examples. These applications essentially combine very large bitmap images, high-quality text and 2D vector graphics, e.g., satellite imagery augmented with data layers, or information visualization techniques for the display of large datasets, e.g., for the visual exploration of large networks (Figure 1-a). However, there is currently no good solution for the distributed rendering of high-quality 2D graphics on cluster-driven wall displays.

Low-level 3D graphics APIs such as OpenGL are currently the main solution for developing cluster-driven visualizations. They work well for the high-performance visualization of textured 3D scenes, but are ill-suited to programming high-quality 2D graphics interfaces, lacking appropriate support for the management and efficient rendering of text, line styles, arbitrary 2D shapes and WIMP widgets. This was already observed for desktop application programming [4], and remains true for cluster-driven wall-displays. Pixel streaming approaches à la SAGE work well when combining different windows of relatively limited size from different applications, potentially running on different machines. They would however not work for full-screen, highly-dynamic visualizations on ultra-high-resolution displays: updating hundreds of megapixels forming a single coherent image at an interactive refresh rate would require significantly more network bandwidth than is commonly available and would put an extremely heavy load on the node in charge of rendering the image to be streamed.

Rich interactive 2D desktop applications, usually termed post-WIMP applications, are typically developed with structured graphics toolkits [1, 5, 10, 16] that provide useful abstractions on top of low-level APIs. They enable rapid prototyping and development of advanced interactive visualizations. Our goal is to offer a structured graphics toolkit capable of running transparently on cluster-driven wall displays and capable of handling a wide range of input devices and

modalities. From a graphics perspective, this requires hiding the complexity entailed by having to distribute rendering on multiple computers. While our focus is on expressiveness and ease-of-use, we also pay attention to scalability issues, adapting ideas originally developed for efficient distributed 3D rendering to our context, such as the use of a multicast protocol to transmit updates to cluster nodes, and a culling algorithm adapted to zoomable user interfaces. From an input management perspective, this requires going beyond the basic redirection mechanisms found in existing distributed rendering frameworks that only support conventional input devices, i.e., mouse and keyboard operated from the master computer. For now, support for other devices is mostly achieved *via* ad hoc solutions (drivers or libraries) that are strongly integrated and statically linked within applications. This approach is not generic and flexible enough when exploring and prototyping novel interaction techniques [7]. An alternative approach consists in providing high-level abstractions of input modalities that enable association and runtime substitution of devices. It has proven successful in other domains, including physical ubiquitous computing [3], virtual reality (Gadeteer for VR Juggler [6]) and in the more general context of post-WIMP applications (Icon [7], Squidy [12]), and we adapt it to interactive wall displays.

jBricks FRAMEWORK ARCHITECTURE

The framework is essentially composed of two independent modules: one for managing all graphical operations, and one for handling input. The two modules are loosely coupled. They communicate via a dynamic plugin architecture and network sockets using high-level protocols such as OSC. This makes the framework highly flexible: modules can be instantiated multiple times and can run on different nodes.

Structured Graphics

Our goal is to provide an API and feature-set similar to those of desktop structured graphics toolkits [1, 5, 10, 16] while i) hiding the complexity entailed by distributed rendering, ii) promoting ease of learning and ease of use, and iii) enabling code reuse: visualization components initially developed for desktop computers should run on cluster-driven wall dis-

plays with minimal changes to the original application code. With these high-level objectives in mind, we chose to extend an existing structured graphics toolkit rather than start developing a new one from scratch.

We used the open-source ZVTM toolkit [16], that supports most Java2D drawing primitives but offers higher-level abstractions that ease the management and manipulation of graphical objects: rendering is handled in retained mode, meaning that the toolkit retains a complete model of the objects to be rendered. ZVTM follows a monolithic approach, as opposed to a polyolithic one¹. Experience has shown that monolithic approaches are conceptually easier to handle by developers, generate less lines of code and require managing a smaller number of objects [5]; properties that we consider of high importance for rapid UI development.

Featured types of graphical objects include polygons of arbitrary shape, splines, Swing widgets, bitmap images and high-quality text, with support for advanced stroke and fill patterns. Those objects (*Glyphs*) are placed on infinite drawing surfaces (*Virtual Spaces*) that are observed through one or more *Cameras*. A camera renders the objects that lie in its viewing frustum in a *View*, that corresponds to a window on the screen. The toolkit makes it easy to create zoomable user interfaces (cameras can be smoothly panned and zoomed). It supports multiple independent views, as well as *Portals* (views within views) [4], multiple layers within a view (each corresponding to a different camera), as well as a variety of built-in focus+context visualization techniques. Cameras and glyphs can be animated using various pacing functions.

Cluster-based Structured Graphics Rendering

jBricks' extension of ZVTM to render graphics on cluster-driven tiled displays is conceptually straightforward. It takes an approach similar to what *sort-first* algorithms do for parallel rendering of 3D graphics in retained mode: as ZVTM already enables multiple cameras to observe a given virtual space, implementing tiled rendering basically consists in sharing that virtual space between all cluster nodes and setting one camera per display tile. Each camera's viewing frustum is configured so that their juxtaposition forms an overall coherent image from the user's perspective, according to the physical layout of display tiles.

Distributed Virtual Spaces. jBricks adopts a client-server model [15]: as shown in Figure 2, a single instance of the application runs on a *client node*, generating the geometry (populating virtual spaces with glyphs) and distributing it to *render servers* running on *cluster nodes*. Virtual spaces and glyphs contained therein are broadcast to all cluster nodes. They are replicated and kept synchronized as glyphs are added, removed, or have their properties changed. Parallel rendering frameworks for 3D graphics have mainly focused on the visualization of static-geometry models where only the camera(s) are manipulated interactively. The applications that jBricks aims to support typically manage much more dynamic objects, both in terms of geometry and visual

¹Monolithic toolkits primarily use compile time inheritance to extend functionality, while polyolithic toolkits primarily use run-time composition to do so, typically using a scene graph [5].

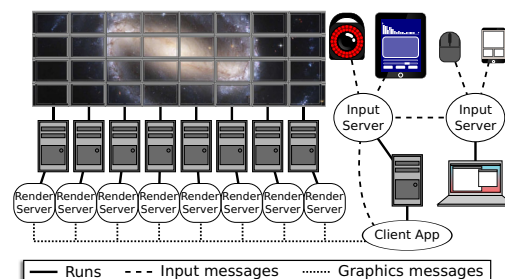


Figure 2. Example jBricks configuration: wall's graphics client and input server for motion tracker and tablet run on client node; input server for mouse, keyboard and smartphone run on user's laptop.

appearance (color, stroke, font, etc.), potentially requiring a lot of network bandwidth. Multicast communication can greatly decrease bandwidth requirements for those updates [13]. We use JGroups (<http://www.jgroups.org>) as our group communication layer, that provides reliable messaging over IP multicast. Over this layer, we exchange atomic changes called *Delta*, which are serialized Java objects representing a new value for a given glyph attribute, propagated to the corresponding glyphs on render servers.

Performance. As noted by Bederson and Meyer [4] about zoomable user interfaces: “*Smooth real-time interaction is crucial. If the system becomes slow and jerky, the metaphor dies*”. The use of a multicast protocol for updating glyphs enables us to smoothly animate several hundred property changes simultaneously and independently of the number of render servers. Camera animations do not require significant bandwidth, as moving a camera only requires updating a maximum of three double-precision floating point values per frame. A more serious bottleneck when panning and zooming is the frame rate achieved by render servers. ZVTM already implements efficient culling algorithms for zoomable user interfaces. Glyphs get projected and rendered for a given camera only if they lie in that camera's viewing frustum. jBricks benefits from this directly: each server renders only the glyphs that will eventually be visible in the associated tile, which significantly decreases the computational and rendering load for scenes with high object counts.

Preliminary tests have shown that visualizations containing up to 200,000 objects could be rendered at interactive frame rates on the platform depicted in Figure 1. jBricks also benefits from Java2D's OpenGL pipeline, and from ZVTM's spatial indexing and dynamic external resource (un-)loading mechanisms. These were developed to support multi-scale navigation in very large datasets, such as gigapixel bitmap images decomposed recursively as a region quadtree. We adapted these mechanisms in jBricks to work in a distributed context, enabling the interactive visualization of very large images. Example images that have been visualized include the 26 gigapixel panorama of Paris (354 048 × 75 520 pixels) and Spitzer's Infrared Milky Way (Figure 1-b), that can be freely panned and zoomed on a wall display.

Programming. jBricks adds cluster support to ZVTM by monkey-patching the original toolkit using AspectJ, with-

out altering its source code. This makes the cluster extension module small ($\approx 3\,000$ lines of code vs. $\approx 39\,000$ for ZVTM) and facilitates forward compatibility. This also keeps API changes to a minimum: virtual spaces, glyphs, animations and most other constructs are managed through the original ZVTM API; low-level mechanisms for distribution to render servers are hidden from the developer. Only cameras and views get created and managed in a slightly different manner. The tiled display's geometry has to be declared: number of rows and columns, size of each screen (pixels), options such as whether to paint pixels behind the bezels separating the tiles (overlay approach) or ignore them (offset approach). *Clustered Views* replace regular ZVTM views: a clustered view is divided into blocks, each block corresponding to a tile and render server. Render servers can be instantiated multiple times on a single node if that node drives multiple tiles. ZVTM-based desktop applications, originally written to run on single hosts, can be adapted to run on a cluster-driven large displays by changing as little as four lines of code. Render servers are instances of a generic display program that is part of jBricks, meaning that developers only have to modify the client application and do not have to run application-specific code on cluster nodes. This enables a quick development and deployment lifecycle. It is also interesting to note that the client application and render servers can run anywhere, including on the same computer, which facilitates development outside the cluster platform.

Advanced Input & Interaction

Wall-sized displays are often augmented with a complex interactive environment, made of heterogeneous input modalities ranging from actual input devices (e.g., mouse, 6-DOF devices, tablets), to the output of interactive systems used for input (e.g., motion-tracking system software, multi-touch table tracker, mobile device sensors interpreter). jBricks's cluster extension to ZVTM handles all aspects related to graphics distribution and rendering, but supports little beyond basic input redirection for conventional devices. An input management system is required to handle the multiple input channels and to ease their fusion so as to eventually deliver high-level input events to applications, that make the description of complex interaction techniques easier [10].

We identified three main requirements for such an input management system. The system should be able to handle various kinds of distributed input in a *generic* way to allow easy substitution of input modalities, and should provide generic output to several distributed applications, no matter whether they were specifically developed for this platform or not. The system should be *extensible*, making it easy to support new devices and functionalities with re-usable processing functions or interaction techniques. Finally, the system should be *adaptable*, enabling runtime addition of new devices and changes to the input configuration.

With these objectives in mind, we developed the jBricks Input Server (jBIS), the distributed input and interaction management system of jBricks. jBIS is built on top of the FlowStates toolkit [2], that combines the ICon [7] and SwingStates [1] libraries. ICon's dataflow model can handle multiple devices and describe advanced interactions ef-

ficiently [10]. Its visual editor makes it simple to connect them to application input endpoints (Figure 3). SwingStates extends the Java language with state machines and provides a simple yet powerful programming language that simplifies the description of interaction logics on the application side. FlowStates integrates these two models seamlessly: state machines are instantiated as dataflow processing devices that can be graphically connected to input devices or to other state machines in the dataflow configuration.

Input handling. Thanks to the ICon library, the jBricks Input Server has built-in support for various regular and advanced input devices: mouse, keyboard, various tablets, Nintendo Wii remotes, VICON motion-trackers, interactive pens, etc. These input devices are instantiated as dataflow processing devices that can be connected to adapters or application devices through the dataflow editor (see the mouse device in Figure 3). These dataflow components are high-level structured representations of input devices (or classes of input devices) with typed output slots mapped to the various channels of the input device they handle.

We extended ICon to support generic devices through various protocols with specific dataflow devices that can receive and send OSC, Ivy or TUIO messages. This approach provides an implicit way of performing automatic device registration thanks to the addressing mechanism of these protocols: each input source that sends a message addressed to a specific receiving device in a running configuration is implicitly considered. For instance, a jBIS' OSC receiver device can listen to messages addressed to `/jBIS/position` with two arguments, *x* and *y*. This device will then externalize the corresponding output slots. These will be updated each time that a new `/jBIS/position` message is received, wherever it comes from: a smartphone running an application that sends OSC messages from touchscreen events, the tracking software of an interactive table, mouse movements from a laptop running another instance of the jBIS, etc.

Interaction configuration. Input configuration and the lower-level description of interaction techniques (typically the connection to inputs) get specified in jBIS with an ICon dataflow configuration. ICon provides an extensive library of adapter devices, e.g., math or logic operators, control structures, flow control. These can be used to manipulate and transform the raw values of input channels into higher-level data structures (e.g., the mult device in Figure 3). The jBIS built-in library also extends the basic processing devices of ICon with platform-specific ones, adapted to interactive wall-sized displays: for instance, the *pointed tile* dataflow component returns the display tile that is intersected by a 3D vector received as input (typically modeling the user's arm). More than simple low-level processing components, these higher-level devices are close to the re-usable interaction techniques of [10], offering several levels of granularity to the user when building an input configuration.

The jBricks Input Server also includes a plug-in mechanism for the creation of custom dataflow devices with FlowStates [2]: state machines are instantiated as dataflow components, and their transitions are triggered by the connected inputs

(pointing and pan-zoom in Figure 3). Programmers can use this descriptive and straightforward approach to extend the jBIS library and to describe some parts of the interaction logic of an application, or even more generic libraries that can be used with multiple applications running on the platform.

Link with application/visualization software. In jBricks, the higher-level interaction logic (manipulation of objects, graphical feedback) is encoded in the client application (Figure 2) developed with ZVTM. The link between the jBIS and this application can be established in two ways. The first solution consists in using specific dataflow devices in the input configuration to deliver high-level interaction events to the application through a networking protocol such as OSC; the client application interprets these messages and reacts accordingly. The other solution consists in using the plugin mechanism of jBIS to implement application-specific devices that will be instantiated as endpoints of the dataflow. These plugins can define their own protocol to communicate with the client application, or even encapsulate it, enabling direct communication as the client node is running in the same process (same Java Virtual Machine) than jBIS.

Finally, jBIS can be controlled remotely, so that applications can trigger commands (start/stop/change the input configuration) or dynamically install a plugin. Several jBIS instances can run simultaneously, communicating through networking dataflow devices (Figure 2). This modularization, based on the description of partial input configurations, reinforces the flexibility and adaptability of the platform as partial configurations can easily be substituted.

The architecture of jBricks and the resulting development and configuration tools make it possible to develop applications outside the platform, i.e., on a simple laptop, and then deploy and run them on an actual cluster-driven wall display. On the graphics side, changes to the client application are minimal (four lines of code) and can easily be managed using, e.g., command line options or Maven profiles. On the interaction side, the jBricks Input Server makes it easy to dynamically reconfigure and adjust inputs according to available devices and modalities. In the following section, we illustrate these principles with a short scenario showing how jBricks can be used for the prototyping and implementation of interaction techniques for a controlled experiment on a wall-sized display.

jBricks IN ACTION

Abelard and Eloïse need to prepare an experiment to compare one-handed mid-air interaction techniques for selection of very small targets on wall-sized displays. They consider two techniques: a very precise bi-modal pointing technique, and a cursor-centered pan & zoom technique.

They first describe the two techniques with state machines (Figure 3) and plan to implement and configure them as follows. The pointing technique will be operated with a gyroscopic mouse and will feature a coarse mode – i.e., ray-casting – and a precise mode – i.e., relative pointing with a low CD gain. Precise mode will be triggered using the right

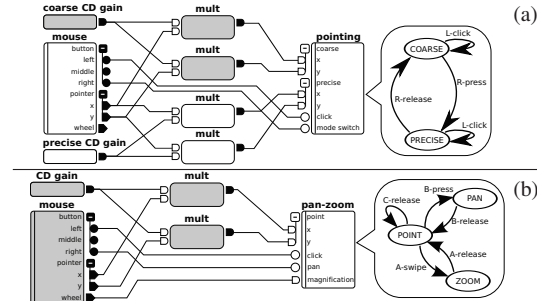


Figure 3. jBIS configurations of the pointing (a) and pan & zoom (b) techniques and their corresponding state machines. A mouse is used to control the techniques and simulate unavailable devices.

mouse button; target selection using the left button (Figure 3-a). The pan & zoom technique is operated with an iPod Touch. Vertical thumb movements control the zoom factor, ray-casting of the user's arm controls the cursor's position. Two small areas at the bottom of the iPod's screen trigger panning and target selection, respectively (Figure 3-b).

Prototyping

As jBricks' graphics and input modules are loosely-coupled, Abelard can work on the experiment's graphics while Eloïse implements and configures the two interaction techniques.

Abelard is working on the graphics part of the experiment. Using ZVTM, he creates an application that displays the targets, cursor appearance and textual instructions on his personal computer without having to worry about the specifics of the cluster-based wall display environment. He just needs to consider the actual dimensions of his graphical scene (in this case, a 20000×7000 pixel area). To make the entire scene visible on his screen, he sets the zoom factor higher than it will eventually be in the real experiment (a straightforward operation in a zoomable user interface).

Meanwhile, Eloïse implements each technique as a Flow-States state machine and encapsulates them in a jBricks Input Server plugin, making them available as dataflow processing devices. During this early prototyping stage, Eloïse focuses on developing the interaction logic, using a basic version of the graphics interface provided by Abelard. She does not need to work on the actual hardware platform either. She runs jBIS on her laptop and uses a regular mouse to simulate the actual input devices that will be used eventually (motion-capture system, gyroscopic mouse, iPod Touch). In this testing configuration, ray-casting with the motion-capture system and gyroscopic mouse are replaced by mouse coordinates; the mouse wheel and buttons are used in lieu of touch events. The output ports of the mouse device are connected to the technique devices, pan-zoom and pointing (Figure 3), the two modes of the pointing technique being simulated by applying constant multipliers to the mouse coordinates (the mul and CD gain processing devices). Later, these configurations will be slightly modified to handle the actual input devices to be used in the experiment.

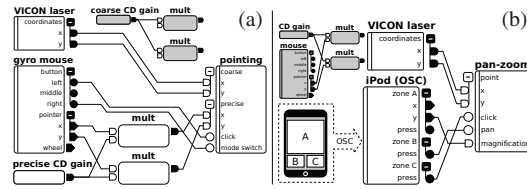


Figure 4. jBIS configurations of the final pointing (a) and pan & zoom (b) techniques. The simulation inputs (in grey) can be reused at any time simply by changing the connections.

Porting to the Wall Display Hardware Platform

On the input side, Eloïse substitutes the devices used for prototyping on her laptop with the platform's actual devices, as shown in Figure 4. The regular mouse can be directly substituted with the gyroscopic mouse, with only a CD gain adjustment (changing the value of precise CD gain, Figure 4-a). jBIS has built-in support for the 10-camera motion tracking system in the room (the VICON laser device). For the iPod Touch, Eloïse uses a built-in OSC receiver device in her input configuration to receive touch events from a freely-available application running on the handheld (Figure 4-b). To deploy the client application on the actual hardware, Abelard only needs to add a few jBricks instructions describing the *Clustered View*. He then embeds the application into the jBIS plugin made by Eloïse. The client application is launched by jBIS; it has access to the state machines' output and will react according to the chosen interaction technique.

Further iterations, switching back and forth between the simplified configuration running on personal computers and the one for the actual wall display hardware is straightforward. Abelard and Eloïse can also easily add new techniques by implementing new state machines and test several input configurations for each of them.

CONCLUSION

The jBricks framework extends and integrates state-of-the-art structured graphics and input management toolkits to enable the rapid development of post-WIMP applications for cluster-based wall displays equipped with advanced input devices and modalities. Its architecture and features enable easy deployment and reconfiguration, allowing developers to partially implement and debug their applications on conventional hardware such as a single laptop or workstation.

We have successfully used jBricks for the rapid prototyping of novel interaction and visualization techniques, and to run controlled experiments for their evaluation [14]. It is also used for the development of various applications for the visualization of large datasets in other disciplines: astrophysics, social network analysis, geospatial intelligence. The Java-based platform makes it easy to use existing libraries in client applications. In addition, ZVTM features several extension modules that enable, e.g., the layout of large networks, the visualization of treemaps, native high-quality PDF rendering, FITS astronomy image display, interactive navigation in OpenStreetMap, from world overview down to street level. Future work will focus on improving the Java2D/OpenGL rendering pipeline by optimizing the

stream of instructions. The implementation of a higher-level communication protocol, based on HID definitions on top of OSC, will improve dynamic input device registration and configuration. jBricks will be made available under an open-source software license (<http://insitu.lri.fr/jBricks>).

ACKNOWLEDGEMENTS

We wish to thank Caroline Appert and Olivier Chapuis for helpful comments on early drafts of this paper. This work is supported by a Région Île-de-France / Digiteo grant.

REFERENCES

1. C. Appert and M. Beaudouin-Lafon. SwingStates: Adding state machines to Java and the Swing toolkit. *SP&E*, 38(11):1149–1182, 2008.
2. C. Appert, S. Huot, P. Dragicevic, and M. Beaudouin-Lafon. FlowStates: Prototypage d'applications interactives avec des flots de données et des machines à états. In *Proc. IHM '09*, 119–128. ACM, 2009.
3. R. Ballagas, M. Ringel, M. Stone, and J. Borchers. istuff: a physical user interface toolkit for ubiquitous computing environments. In *Proc. CHI '03*, 537–544. ACM, 2003.
4. B. Bederson and J. Meyer. Implementing a zooming user interface: experience building pad++. *SP&E*, 28:1101–1135, August 1998.
5. B. B. Bederson, J. Grosjean, and J. Meyer. Toolkit Design for Interactive Structured Graphics. *IEEE Trans. Software Eng.*, 30(8):535–546, 2004.
6. A. Bierbaum, C. Just, P. Hartling, K. Meinert, A. Baker, and C. Cruz-Neira. VR Juggler: A Virtual Platform for Virtual Reality Application Development. In *Proc. VR '01*, 89. IEEE, 2001.
7. P. Dragicevic and J.-D. Fekete. Support for input adaptability in the icon toolkit. In *Proc. ICMI*, 212–219. ACM, 2004.
8. S. Eilemann, M. Makhinya, and R. Pajarola. Equalizer: A Scalable Parallel Rendering Framework. *IEEE TVCG*, 15(3):436–452, 2009.
9. G. Humphreys, M. Houston, R. Ng, R. Frank, S. Ahern, P. D. Kirchner, and J. T. Klosowski. Chromium: a stream-processing framework for interactive rendering on clusters. *ACM Trans. Graph.*, 21(3):693–702, 2002.
10. S. Huot, C. Dumas, P. Dragicevic, J.-D. Fekete, and G. Hégron. The MaggLite post-WIMP toolkit: draw it, connect it and run it. In *Proc. UIST '04*, 257–266. ACM, 2004.
11. B. Jeong, L. Renambot, R. Jagodic, R. Singh, J. Aguilera, A. Johnson, and J. Leigh. High-performance dynamic graphics streaming for scalable adaptive graphics environment. In *Proc. SuperComputing*. ACM, 2006.
12. W. König, R. Rädle, and H. Reiterer. Interactive design of multimodal user interfaces. *J Multimod. UI*, 3:197–213, 2010.
13. M. Lorenz, G. Brunnett, and M. Heinz. Driving tiled displays with an extended chromium system based on stream cached multicast communication. *Parallel Comput.*, 33(6):438–466, 2007.
14. M. Nancel, J. Wagner, E. Pietriga, O. Chapuis, and W. Mackay. Mid-air pan-and-zoom on wall-sized displays. In *Proc. CHI '11*. ACM, 2011. In press.
15. T. Ni, G. S. Schmidt, O. G. Staadt, M. A. Livingston, R. Ball, and R. May. A Survey of Large High-Resolution Display Technologies, Techniques, and Applications. In *Proc. VR '06*, 223–236. IEEE, 2006.
16. E. Pietriga. A Toolkit for Addressing HCI Issues in Visual Language Environments. In *Proc. VL/HCC '05*, 145–152. IEEE, 2005.

Résumé en Français

Les récentes avancées en matière d'acquisition, stockage et traitement des données ont résulté en une augmentation spectaculaire de la quantité d'information collectée et analysée dans de nombreux domaines : disciplines scientifiques, industrie, commerce, bibliothèques numériques, données mises à disposition du public par des organismes gouvernementaux, domaines auxquels s'ajoutent les contenus que tout un chacun peut mettre sur le Web à travers les réseaux sociaux, les blogs ou les sites de partage de documents multimédia. La capacité des systèmes et bases de données hétérogènes d'échanger et croiser ces données, mais aussi d'inférer de nouvelles données, s'est aussi grandement améliorée, grâce à des technologies émergentes comme celles issues des recherches liées au Web des données [BL09] (par opposition au Web des documents) s'appuyant sur les technologies du Web sémantique.

Ces technologies, qui vont des services Web aux ontologies et associent les données semi-structurées [ABS99] à des informations sémantiques interprétables par les machines, pourraient révolutionner une nouvelle fois les activités dans de nombreux domaines, de plus en plus centrées sur les données. Grâce à ces technologies, des systèmes hétérogènes et distribués à l'échelle d'Internet peuvent échanger de l'information, produire automatiquement de nouvelles données par le biais de moteurs d'inférence et des ontologies associées aux données de base, et croiser des sources de données multiples grâce à des mécanismes de résolution de conflits. Les jeux de données résultants sont souvent très grands, et peuvent ensuite être interconnectés comme c'est le cas par exemple dans le cadre de l'initiative *Linked data*.

La gestion de quantités massives de données pose des problèmes de recherche dans de nombreuses spécialités du domaine de l'informatique : représentation des connaissances et bases de données, communication & réseaux, sécurité, fouille de données, mais aussi interprétation et manipulation de ces données par leurs créateurs et utilisateurs. Mes travaux de recherche s'inscrivent dans le domaine de l'interaction homme-machine (IHM), et portent plus spécifiquement sur la conception, le développement et l'évaluation de techniques d'interactions et de visualisation facilitant la compréhension et la manipulation de ces masses de données.

Le but des recherches en IHM est de rendre les ordinateurs plus faciles à utiliser tout en augmenter les capacités des utilisateurs pour leur permettre de gérer des problèmes plus complexes, des jeux de données plus grands, de manière la plus efficace possible, que ce soit dans des contextes mono-utilisateur ou de travail collaboratif. Une description plus formelle est que l'IHM s'intéresse à la conception de systèmes qui réduisent la barrière entre le modèle cognitif que ce font les humains de la tâche qu'ils doivent accomplir et la compréhension qu'a l'ordinateur de cette même tâche. L'IHM concerne la conception, le développement et l'évaluation des systèmes informatiques avec lesquels des personnes ont à interagir. Il s'agit d'un domaine de recherche multi-disciplinaire, regroupant des experts en informatique, sciences cognitives, design, ingénierie, ethnographie, facteurs humains et sociologie. Dans le contexte plus précis décrit plus haut, la recherche en IHM se concentre sur la conception, le développement et l'évaluation de techniques d'interaction et de visualisation qui aident les utilisateurs à mieux comprendre et manipuler des jeux de données (semi-)structurés plus grands et plus complexes. Les utilisateurs peuvent être de simples consommateurs des données, essayant d'interpréter

l'information et d'en extraire de nouvelles connaissances et idées; ils peuvent aussi être les producteurs de ces données, les créant, les structurant, les transformant et les publiant pour consommation par d'autres utilisateurs. Dans tous les cas, producteurs et consommateurs de données doivent faire face à des grands volumes de données, organisés en structures plus ou moins complexes, et qui peuvent être interconnectées. Quel que soit le niveau de complexité et la richesse, en termes de fonctionnalités, des systèmes d'acquisition, de traitement, et de stockage, les données sont produites par des utilisateurs, et au final exploitées par des utilisateurs.

Ma recherche est basée sur la conviction que l'interaction homme-machine, et plus spécifiquement les interfaces graphiques, peuvent être d'une grande aide aux utilisateurs quand elles fournissent des visualisations pertinentes des données, de leur structure, et qu'elles sont couplées à des techniques d'interaction permettant une navigation efficace dans de grands espaces d'information. Elles jouent à ce titre un rôle important dans la recherche et le développement de systèmes informatiques pour la gestion et l'analyse de masses de données semi-structurées. Mes activités de recherche des dix années passées ont été organisées autour de deux thèmes principaux, présentés en détail dans ce mémoire :

- langages visuels et techniques de visualisation d'information pour aider les utilisateurs à interpréter et manipuler des jeux de données semi-structurés ;
- conception, développement et évaluation expérimentale de techniques d'interaction multi-échelle pour naviguer dans des masses de données.

Visualisation et transformation de données structurées

Les langages de description de données semi-structurées, des couches basses comme XML aux descriptions ontologiques situées à un plus haut niveau d'abstraction permises par OWL, permettent aux utilisateurs d'organiser leur données. Elles permettent aussi aux machines d'effectuer automatiquement des traitements élaborés sur ces données en leurs associant une sémantique que les ordinateurs sont capables d'interpréter: au-delà de la structuration des données, des vocabulaires permettent de décrire les données grâce à des vocabulaires ou des ontologies, d'interconnecter les jeux de données, et de les croiser.

Le Web que nous utilisons depuis de nombreuses années permet essentiellement à des utilisateurs humains de consulter des pages Web. Ce Web est maintenant souvent appelé *Web des documents*, pour le différencier du *Web des données* reposant sur les technologies mentionnées ci-dessus. Le Web des données permet de partager de l'information sous forme de données, provenant de différentes sources qui peuvent être lues automatiquement, et qui peuvent dans une certaine mesure servir de base à des raisonnements automatiques effectués par les machines. Les langages sous-jacent, XML inclus, sont conçus pour faciliter l'interprétation des données par les machines, à l'opposé des pages HTML du Web des documents, dont les principaux consommateurs sont des lecteurs humains.

La plupart des utilisateurs finaux du Web des données ne verront jamais de lignes de code XML ou RDF. Mais certaines catégories d'utilisateurs sont confrontés aux données brutes et doivent les comprendre et les manipuler :

- experts dans un domaine particulier qui créent les vocabulaires et les ontologies qui associent aux données une sémantique exploitable par les machines, et les peuplent avec des données réelles ;
- développeurs qui écrivent des programmes pour interroger et manipuler les données, par exemple pour établir des corrélations, inférer de nouvelles informations, mais aussi au final pour transformer les structures de données résultant de ces processus vers un vocabulaire cible comme HTML afin d'être lues et visualisées par tout type d'utilisateur.

Mes travaux de thèse de 2000 à 2002 au Xerox Research Centre Europe et à l'INRIA Rhône-Alpes ont porté sur la conception et l'implémentation d'un langage de programmation visuelle, VXT, pour la spécification de transformations de documents XML [PVDQ01, PVD01]. Le but de ce langage était de faciliter le développement de transformations XML en tirant parti du caractère multi-dimensionnel des langages visuels pour représenter explicitement la structure hiérarchique des données, considérée comme un élément d'information à part entière, que même les langages situés à un haut niveau d'abstraction comme XSLT ne peuvent pas montrer, en partie à cause de leur caractère uni-dimensionnel. Une partie importante de mon travail était alors concentrée sur la définition formelle de la syntaxe et de la sémantique du langage et sur l'établissement de preuves quant à sa complétude et à la validité de ses productions [Pie02a, VDP01]. Je me suis cependant aussi beaucoup intéressé aux problématiques d'IHM liées à la conception de l'environnement de développement pour ce langage visuel. Les langages visuels peuvent être des outils puissants, et certains comme LabView [Pow11] sont eu beaucoup de succès. Mais ils requièrent une attention particulière en termes de conception de l'interface graphique et des interactions permettant de manipuler les structures programmatiques du langage et les données [GP96]. Parmi les problèmes principaux se trouve celui du passage à l'échelle [BBB⁺95]. Cela était particulièrement vrai pour VXT, qui devait représenter visuellement des structures de données hiérarchiques potentiellement très grandes et complexes, les grammaires structurelles associées, et les règles de transformations de type XSLT. Ceci m'amena à adopter des techniques récentes du domaine de la visualisation d'information, notamment pour la représentation de structures arborescentes et pour la navigation multi-échelle dans de grands espaces d'information.

Une visite dans l'équipe W3C du MIT à l'automne 2001 me donna la possibilité de travailler sur un problème proche : celui de concevoir une solution d'édition graphique interactive de données RDF [Pie02b]. La structure de ces données, de type graphe étiqueté et orienté, est très difficile à percevoir à travers les représentations textuelles. Même si ces dernières sont le moyen principal de représentation et d'échange d'information entre agents logiciels, elles sont loin d'être optimales quand il s'agit d'échanger l'information entre le système informatique et l'utilisateur. Les représentations visuelles peuvent aider, mais introduisent de nouveaux problèmes, et mon travail a consisté à proposer des solutions à cer-

tains de ces problèmes. Ce travail, continué en post-doctorat dans le groupe DIG au MIT en 2003 [Pie03, KKPS03, Pie06], et la collaboration qui y fit suite autour du langage Fresnel en 2005-2006 [BLP05, PBKL06, PL09], sont détaillées dans la Chapitre 2, de même que des projets collaboratifs plus récents, dans lesquels ma contribution a essentiellement porté sur la conception et l'implémentation de visualisations interactives de données semi-structurées pour des applications dans des domaines spécifiques [BLP10, BPLL11, MCH⁺09].

Navigation multi-échelles dans des masses de données

Après cette année de post-doctorat et une autre passée dans l'industrie, j'ai rejoint l'équipe-projet In-Situ à l'INRIA Saclay comme Chargé de Recherche, à l'automne 2004. Mon intérêt pour l'IHM et plus particulièrement pour les techniques d'interaction et de visualisation d'information s'était amplifié au cours des années, et je souhaitais mener des recherches dans ce domaine. In-Situ m'a fourni d'excellentes conditions pour effectuer cette transition.

Mes travaux dans ce domaine, décrits dans le Chapitre 3, ont été essentiellement centrés sur la navigation multi-échelle dans des données massives, représentant des données semi-structurées ou d'autres types de données tels que des réseaux [MCH⁺09] ou des données provenant par exemple systèmes d'information géographiques. D'abord par la définition d'une méthode d'opérationnalisation des tâches de recherche multi-échelle permettant d'évaluer les performances de techniques d'interaction [PAB07]. Puis par une étude approfondie des techniques de représentation dites *focus+context* qui grossissent localement une zone d'intérêt d'un grand espace d'information pour fournir plus détails sur les éléments qu'elle comporte. Les techniques de ce type utilisaient jusqu'alors des distorsions spatiales pour intégrer la zone d'intérêt de manière continue dans son contexte, ce qui posait des problèmes d'interprétation de la visualisation et de performance d'acquisition de cible. Nous avons dans un premier temps proposé des alternatives à la distorsion spatiale, fondées sur la dimension temporelle et sur le contrôle fin de la semi-transparence [PBA10, PA08]. Nous nous sommes ensuite intéressés au problème de *quantization* posé par les forts facteurs de grossissement lors des tâches d'acquisition de cibles [ACP10], associé à la problématique des techniques facilitant le pointage de petites cibles sur laquelle j'ai collaboré avec d'autres membres de l'équipe [CLP09].

La visualisation multi-échelle de données massives est aussi le problème de recherche au centre du projet *WILD*, mené en collaboration avec le LIMSI-CNRS et l'équipe-projet Aviz (Analyse Visuelle) à l'INRIA Saclay. Le but de ce projet, que je coordonne avec Michel Beaudouin-Lafon, et qui a représenté l'essentiel de mon activité de 2008 à 2010, est d'étudier les dispositifs d'affichage ultra-haute-résolution pour la visualisation collaborative de masses de données scientifiques. La première phase a consisté en la construction d'une plate-forme de visualisation composée d'un mur d'écrans de 5.5m de large par 1.8m de haut offrant une capacité d'affichage de 131 millions de pixels, piloté par une grappe de seize machines, couplé à un système de capture de mouvements et à des surfaces tactiles (tables interactives, smartphones, tablettes). La deuxième phase du projet consiste en trois types d'activité :

- La conception et l'évaluation de techniques d'interaction et de visualisation génériques optimisées pour ce type de plate-forme, les dispositifs d'entrée conventionnels type souris+clavier n'étant pas adaptés dans ce contexte. Ce sujet de recherche a donné lieu à mon premier co-encadrement d'étudiant en thèse, Mathieu Nancel, débuté en octobre 2008 avec Michel Beaudouin-Lafon, et a abouti à des publications majeures pour l'équipe, dont [NWP⁺11].
- La conception et l'implémentation d'applications de visualisation en collaboration avec des chercheurs de différents laboratoires de l'Université Paris-Sud et du plateau de Saclay, notamment l'Institut d'Astrophysique Spatiale (visualisation d'images FITS – observations du satellite Planck), le Laboratoire de l'Accélérateur Linéaire (visualisation de résultats d'expérience en 3D) et le laboratoire Neurospin (visualisation comparative de scans de cerveaux).
- Le développement de boîtes à outils facilitant le prototypage des techniques d'interaction et le développement des applications mentionnées ci-dessus. Ainsi, la boîte à outils ZVTM mentionnée plus haut a été adaptée pour fonctionner sur des grappes d'ordinateur, permettant aux applications qui l'utilisent d'être affichées sur des stations de travail conventionnelles comme sur des murs d'écran tels que WILD, sans déploiement spécifique sur la grappe [PHNP11].

En parallèle à ces travaux sur la navigation multi-échelle, mes travaux précédents sur la visualisation de données structurées (réseaux/graphes) m'ont permis d'initier plusieurs collaborations, dont une commencée début 2011 avec le *Japan Advanced Institute of Science and Technology*, portant sur la modélisation et la simulation du comportement d'entités biologiques tels que les virus type λ phage. Enfin, une collaboration ambitieuse a démarré fin 2009 avec l'ESO (*European Southern Observatory*) autour des interfaces de *monitoring* et de contrôle du radio-télescope ALMA (*Atacama Large Millimeter/submillimeter Array*). Ce partenariat de plusieurs années entre ALMA et l'INRIA, mené en étroite collaboration avec les chercheurs et ingénieurs de l'ESO et du NRAO (*National Radio Astronomy Observatory*), a pour but de concevoir et de développer de nouvelles interfaces homme-machine pour le contrôle du télescope adaptées à la complexité de l'instrument, fondées sur des composants de visualisation avancés de données temporelles, de réseaux denses et d'informations géographiques intégrés dans une interface de navigation multi-échelle. Cette collaboration va continuer au sein du CIRIC (*Communication and Information Research and Innovation Center*), centre de recherche que l'INRIA a ouvert au Chili en mars 2012 en partenariat avec six universités.

Perspectives

En juillet 2012, je rejoindrai le CIRIC. Ma mission, d'une durée initiale de deux ans, sera de monter une équipe dont les thèmes seront centrés sur la visualisation de masses de données en temps réel. Cette équipe travaillera en partenariat avec des chercheurs de l'*Universidad de Chile* ainsi qu'avec des entreprises et organismes comme ALMA, la collaboration avec ce dernier venant naturellement s'inscrire dans les thèmes de recherche de cette équipe.

L'équipe développera aussi sa propre plate-forme de visualisation interactive de masses de données sur grandes surfaces d'affichage à densité de pixels élevée, ce thème de recherche étant encore relativement peu exploré et présentant de nombreuses opportunités en termes de recherche et de collaboration à la fois académiques et industrielles, par exemple pour la visualisation d'infrastructures réseaux (les réseaux de communication étant un des axes de recherche principaux du CIRIC) ou du fonctionnement des exploitations minières du pays.

Cette orientation de mes travaux vers la visualisation interactive de masses de données en temps réel a pour but, à terme, de développer mon activité de recherche plus spécifiquement sur l'interaction homme-machine pour les systèmes critiques, type salle de contrôle de grands équipements ou centres de gestion de crise, dans lesquels les systèmes d'information doivent fournir aux utilisateurs les données les plus pertinentes, les aider à les interpréter, afin de fournir un support de décision optimale dans un contexte distribué.