Mélange: Space Folding for Visual Exploration

Niklas Elmqvist, *Member, IEEE*, Yann Riche, Nathalie Henry, and Jean-Daniel Fekete, *Member, IEEE*

Abstract—Navigating in large geometric spaces—such as maps, social networks, or long documents—typically require a sequence of pan and zoom actions. However, this strategy is often ineffective and cumbersome, especially when trying to study and compare several distant objects. We propose a new distortion technique that folds the intervening space to guarantee visibility of multiple focus regions. The folds themselves show contextual information and support unfolding and paging interactions. We conducted a study comparing the space-folding technique to existing approaches, and found that participants performed significantly better with the new technique. We also describe how to implement this distortion technique, and give an in-depth case study on how to apply it to the visualization of large-scale 1D time-series data.

Index Terms—Interaction, visualization, navigation, exploration, folding, split-screen, space distortion, focus+context.

1 Introduction

Current visualization applications often involve navigation in large visual spaces—many times the size of the screen—using a sequence of zoom and pan operations. The tasks that are performed in these spaces typically require multiple objects to be displayed at sufficient scale for precise manipulation, yet these objects may be separated by long distances. Zooming and panning is tedious, potentially disorienting, and often ineffective [11, 12]. In order to retain view of these multiple objects, the standard practice is to split the screen into several subwindows [37], but this may cause context between objects to be lost.

Consider a researcher planning a conference trip to the CHI 2008 conference from eastern Canada to Florence, Italy. Beyond constraints such as cutting costs and minimizing the number of stops and the flight time, the researcher may be interested in combining such a long trip with visits to other labs in Europe. Thus, our traveler wants to study maps of both the source and destination areas at sufficiently high detail to make informed decisions about departure and arrival airports as well as appropriate ground transportation and lodging, yet is also interested in seeing the context between these areas to get an idea of opportunities for potential detours and research visits. Panning and zooming the map to solve this task is burdensome and ineffective. Similarly, splitting the screen to show several regions of the map simultaneously causes loss of context of the intervening space. Figure 1(a) illustrates how the Mélange technique presented in this paper solves this problem by folding the intervening space into the depth of the screen.

The situation is very similar when exploring social networks. These can be represented as matrices to avoid node overlap or edge crossings, which is particularly useful for dense and large networks [14]. Here, nodes are placed on the row and column axes, and a filled cell in the matrix indicates an edge between nodes. Often, several different parts of the same matrix are interesting for a particular task, such as collaborating actors, as well as the intermediate context between them (the communities they belong to). However, no efficient technique exists for studying these parts simultaneously, leaving the user no option but to pan and zoom to navigate. Figure 1(b) shows our approach.

The above two scenarios are examples of extended multi-point interaction tasks [37]—we thus call them *multi-focus interaction* tasks—that require several concurrently visible focus points as well as knowledge of the surrounding and intervening space. We know of no existing presentation technique that fulfills all of these requirements. Therefore, we present the *Mélange* technique that automatically folds away intervening space between focus regions to guarantee their visibility.

An earlier version of this paper was presented at the ACM CHI 2008 conference in Florence, Italy [9]. Compared to that version, the present article incorporates the consolidated results and feedback we have received since the conference presentation. In particular, this article includes details on designing and implementing Mélange for both 1D and 2D spaces, and gives a case study of how to apply the technique to the visualization of a large time-series dataset. We also introduce a semantic layer for non-distorted annotations, labels, and graphics.

The rest of this paper is structured as follows: We begin with a background on exploring visual spaces and multi-focus interaction. We then present the Mélange technique. This is followed by extensive notes on how to implement Mélange in both 1D and 2D. We describe our controlled experiment and present the results and a discussion of our findings. We conclude with a case study showing how to use Mélange for navigation in a large 1D time-series dataset.

2 Background

Visual exploration is defined as the use of visualization to form, test, and validate hypotheses about complex or large datasets [21]. In this work, we consider visual exploration of large, geometric 1D or 2D spaces, such as maps, graphs, or social networks.

2.1 User Tasks

In the context of large visual spaces, we define the general *space exploration* task as a directed search for specific targets while retaining awareness of the context and the overall structure of the visual space. In other words, in this paper we do not consider unstructured exploration where the objective is merely to learn the features of the space.

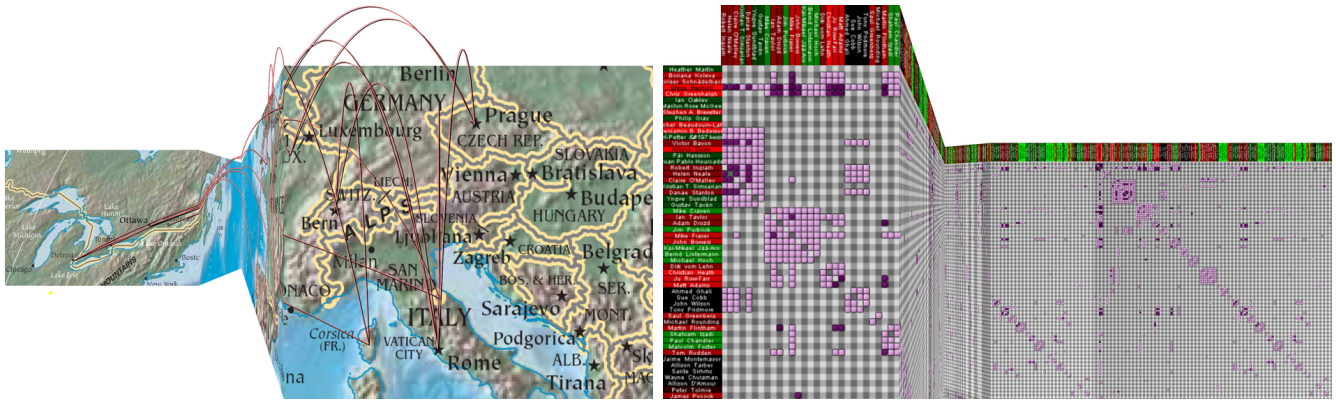
More specifically, our work deals with comparison and correlation between two or more foci on the visual space, a compound user task [1]. Because the visual spaces that we are dealing with are easily larger than the size of the viewport, often hundreds or thousands of times more so, any two or more targets that the user wants to compare are likely to be distant from each other. This means that they will not all fit on the same screen while being displayed at a useful magnification level. Previous work on multi-point interaction [37] has dealt with this problem using multiple fisheye lenses and split-screen techniques.

In our generalized *multi-focus interaction* task, we also stipulate that each focus must be independently zoomed so that the user can adapt the magnification to the task. Furthermore, as much display space as possible should be dedicated to each focus to show its surrounding context. Finally, our intended user tasks often require an awareness of

- *Niklas Elmqvist is with Purdue University in West Lafayette, IN, USA, E-mail: elm@purdue.edu.*
- *Yann Riche is an independent researcher in Seattle, WA, USA, E-mail: yann.riche@gmail.com.*
- *Nathalie Henry is with Microsoft Research in Seattle, WA, USA, E-mail: nathalie.henry@gmail.com.*
- *Jean-Daniel Fekete is with INRIA in Paris, France, E-mail: jean-daniel.fekete@inria.fr.*

This is an extended version of a paper that appeared at the ACM CHI 2008 Conference on Human Factors in Computing Systems, held April 5–10, 2008 in Florence, Italy [9].

Submitted to IEEE Transactions on Visualization and Computer Graphics. Do not redistribute.



(a) Browsing flight routes on a world map.

(b) Displaying a large matrix visualization of a social network.

Fig. 1. Examples of applying the Mélange technique to (a) maps and to (b) social networks.

the content and quantity of space that lies between the foci. For the world map example (Figure 1(a)), context and distance helps the user quickly estimate flight time and stopovers on the way. For the social network (Figure 1(b)), they give an indication of the global communities and collaboration patterns.

2.2 Design Goals

Based on the above requirements for multi-focus interaction on large visual spaces, we formulate a number of design goals for our approach to visual exploration of such spaces:

G1 *guaranteed focus visibility:* multiple foci at independent zoom levels should be visible simultaneously, regardless of their location on the space;

G2 *surrounding context visibility:* as much as possible of the area surrounding each focus region should be visible;

G3 *intervening context awareness:* the space between focus regions should be shown to give a frame of reference; and

G4 *distance awareness:* some notion of the distance between the focus regions should be available.

2.3 Existing Techniques

There are a number of existing techniques (or combinations of techniques) that partially fulfill the design goals outlined above. The rest of this background section reviews the main approaches:

- *General navigation:* interaction techniques for navigating in large visual spaces;
- *Split-screen:* dividing the viewport into smaller subwindows, each showing a small region of the space;
- *Space distortion:* deforming geometric space; and
- *Semantic distortion:* deforming semantic space.

Table 1 gives a summary of these strategies and their properties.

Solution strategy	G1	G2	G3	G4	Techniques
General navigation	–	–	–	–	[2, 19]
Split-screen	Y	Y	–	–	[37]
Fisheye views	Y	P	Y	–	[11, 8, 37]
Rubber sheet	P	P	Y	–	[36, 24, 38]
Semantic distortion	Y	Y	Y	–	[7, 29]

Table 1. Design goals fulfilled by existing strategies (P = partially).

2.4 General Navigation

Zooming and panning are the standard actions for interacting with large visual spaces that exceed the size of the viewport. Furnas and Bederson present the space-scale diagram [13] as a comprehensive model for describing these actions as paths through scale-space. Combining zooming and panning is both more efficient and more informative than using just panning [6, 13, 39]. However, mere zoom and pan operations do not directly support any of our design goals.

A number of approaches have been developed to better support navigation in zoomable spaces. Speed-dependent automatic zooming [19] (SDAZ) seamlessly zooms out to maintain a fixed visual flow depending on the speed of scrolling. Bourgeois and Guiard [6] show that bimanual multi-scale navigation outperforms standard navigation. OrthoZoom [2] allows for controlling both zoom and pan using the orthogonal axes of the mouse in a 1D scrolling task, and was recently shown to be the currently fastest one-dimensional scrolling technique.

For larger visual spaces, standard navigational aids include an overview window showing the position and general context of the viewport on the canvas [28]. A recent trend integrates the overview in the detail view to provide off-screen target awareness; examples include Halo [3], where circles emanating from off-screen targets indicate their approximate distance and location, City Lights [41] that show the "shadows" of off-screen targets on window borders, and the EdgeRadar [15] that provides a rectangular context region on window edges. Hopping [20] extends the idea by also allowing for direct teleportation to any of the off-screen targets indicated on the viewport edge. However, again, these techniques do not provide multiple foci, and give poor support for context awareness.

2.5 Split-Screen

Splitting the screen into several windows showing different parts of the visual space is a standard method employed by commercial applications such as Microsoft Excel and Adobe Photoshop. However, we know of no evaluation on navigation performance in such setups.

Shoemaker and Gutwin [37] present an interaction technique called split-scrolling that automatically divides the screen into two viewports when two interaction points move apart, but they neither implement nor empirically evaluate this technique.

For time-series data, it is useful to be able to summarize or condense periods of times into aggregated representations. An example is LifeLines [30], where the time navigation scrollbar can be split into several regions with multiple foci.

By definition, split-screen setups support the guaranteed visibility (G1) and surrounding context (G2) goals, but intervening context (G3)

and distance (G4) is lost. Adding an overview helps to show the context, but overviews are typically small and placed in the periphery of the viewport, dividing the user's attention between focus and overview, and consuming screen real estate. Nevertheless, split-screen is the most common and most straightforward approach to multi-focus tasks.

2.6 Space Distortion

Space-distortion techniques non-linearly deform the space to optimize navigation. Fisheye views [11, 12] provide ways of doing this both in geometric as well as data space. The Table Lens [31] shows how to apply fisheye distortion to a tabular visualization. The Document Lens [35] visualizes a large document as a rectangular array of pages with a focused region in 3D. This use of 3D perspective foreshortening as a distortion technique is also employed by the Perspective Wall [23]. However, most of these approaches do not directly support our design goals, although they can be used as starting points for fulfilling them.

The rubber sheet stretching metaphor [36] is a model for distorting 2D space. Accordion Drawing [24] (AD) is an extension of the rubber sheet with support for guaranteed visibility. Slack et al. [38] present a general application framework for accordion drawing. The AD method supports all of our design goals, but some of them only partially. Focus regions cannot be zoomed independently (G1), the model is not view-dependent so surrounding context is not automatically allocated a maximum amount of space (G2), and the compressed space gives no direct distance awareness (G4).

Instead of distorting the whole space, Shoemaker and Gutwin [37] describe a multi-point interaction technique based on automatic creation of fisheye lenses for each focus point. Like for the AD method, this approach supports design goals G1 and and G3, but there is no automatic space allocation given the available space (G2), and distance awareness (G4) is difficult to attain when the space is non-linearly deformed. For our multi-focus task, it makes more sense to deform the context regions and leave the focus unchanged at normal size, whereas fisheye lenses allocate space for the foci and leave the context unchanged.

Unifying rubber sheet methods with general space distortion, Carpendale and Montagenese [8] present a comprehensive approach called Elastic Presentation Framework (EPF) for both distorted and non-distorted 2D presentations. Just like for Mélange, EPF conceptually places 2D information spaces on a plane in three-dimensional space—essentially, a geometric interpretation of space-scale diagrams [13]. Zanella et al. [40] later evaluated the effect of visual cues on perceiving space distortion in a study not unrelated to ours.

2.7 Semantic Distortion

As stated earlier, fisheye views [11] also allow for semantic zooming [27], e.g. distorting semantic space instead of geometric space. DOITree [7] and SpaceTree [29] are examples of such techniques for hierarchical structures. However, while this approach can support design goals G1 through G3, it is again distance awareness (G4) that is lacking due to the scale-independent graphical representation.

3 Mélange: Folding 1D or 2D Space into 3D

Mélange is a space deformation technique that folds 2D space into 3D in order to bring several focus regions of interest into view at the same time. Figure 1 shows a large world map being folded using Mélange to bring both northern Italy and eastern Canada into view at high magnification, as well as a matrix visualization of a social network being folded to simultaneously view different parts of the network.

In the following sections, we discuss how Mélange supports all four of the design goals listed in Section 2.2. We also discuss some of the decisions involved in designing the technique.

3.1 Guaranteed Focus and Context Visibility

Given a set of focus points and the location and extents of the current viewport on the canvas, the objective of the Mélange technique is to combine different parts of the visual space so that the focus points and as much as possible of their surrounding context are visible on the user's screen. This fulfills the *guaranteed focus visibility* (G1) and *surrounding context visibility* (G2) design goals.

Focus points are specified as 2D positions on the visual space, and also have an associated depth parameter that allows each point to be zoomed independently of the others. This supports interactions where different parts of the visual space must be viewed at different scales, such as a social scientist studying a particular actor in relation to a larger clique of actors on a matrix representation of a social network.

3.2 Intervening Context Awareness

Using split screens to show multiple foci would remove space outside of the focus regions and show each region as small subwindows in the main viewport. Mélange instead *folds* the space into the negative depth dimension (i.e. into the screen, see Figure 1). If there is no extraneous space to fold away, the space is instead stretched, similar to the rubber sheet [36] but with support for independent depths for each focus point (in other words, folding in Mélange is equivalent to compressing extra space in the rubber sheet, and stretching small space is the same).

The folds themselves are shown in 3D perspective as they stretch away into the depths of screen, and they also indicate the relative positioning of the focus points. This fulfills the *intervening context awareness* (G3) design goal. Furthermore, the mechanism gives a tangible and compelling metaphor for the user that is close to how real paper or fabric is folded. We believe that this metaphor is easier to understand than merely compressing the space, as in rubber sheet models [24, 36].

Figures 2 and 3 show schematic overviews of the folding process for 1D and 2D spaces, respectively. The user's viewport (denoted by the smaller rectangle in the left part of the figure) is centered on the focus point A—the main focus—but the user has also designated a second focus point, B (in both cases at the same zoom level as A). Given the available space in the viewport, the Mélange technique folds away some of the intervening space between the focus points so that also B is visible on the screen. All folds are rectilinear to simplify understanding of the deformed space. A certain amount of screen real estate (*foldSize*) is used to show the contents of the folded space in 3D perspective as it stretches away into the depths of the screen. These regions serve as context between the focus points.

The above method generalizes to any number of additional focus points. One of the foci is always designated as the main one and is used as a baseline for computing the size allocations for the others.

3.3 Context and Distance Awareness

Deforming space to bring several foci onto the screen may cause inaccurate perception of the size of the visual space. For example, folding a world map to bring London and Boston into focus at high detail level will certainly convey a false sense of the distances between the cities.

Mélange supports better *distance awareness* (G4) than compression-based techniques (like the rubber sheet [36]) since the 3D perspective of the folds gives an indication of the distance between the regions.

To further improve distance awareness, we introduce fold pages and interaction techniques for flipping between them. The folded space is split by a suitable and tangible unit, such as the size of the screen. Only one such unit is shown at full detail, and the rest are shown as thin fold pages (Figure 4). Each fold page represents one screen of compressed space. This helps fulfill the distance awareness (G4) design goal by allowing the user to quickly estimate the number of fold pages to find the distance (like estimating a book's length from its thickness).

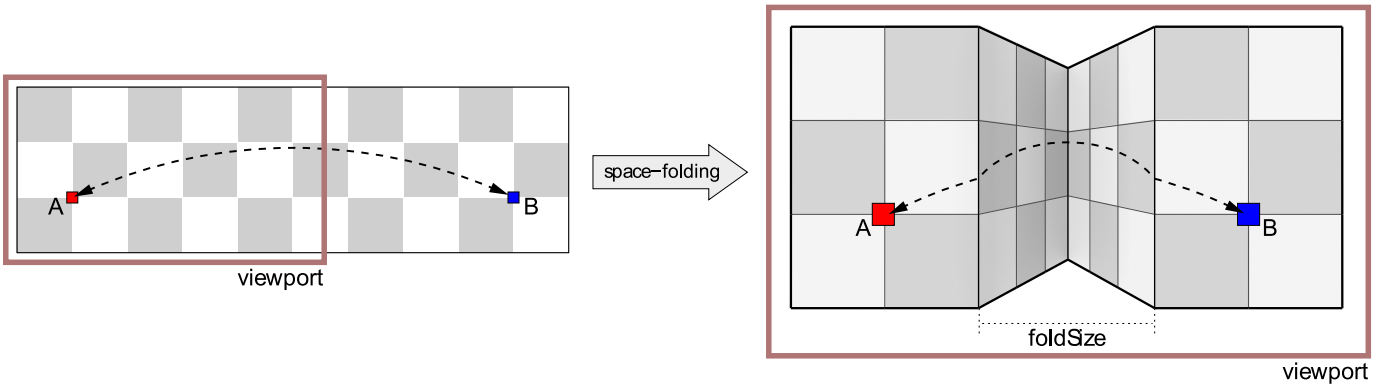


Fig. 2. Folding a 1D space with two focus points A (main) and B. The space is folded to make best use of the available area in the viewport. Focus points can be independently zoomed by changing their 3D depths.

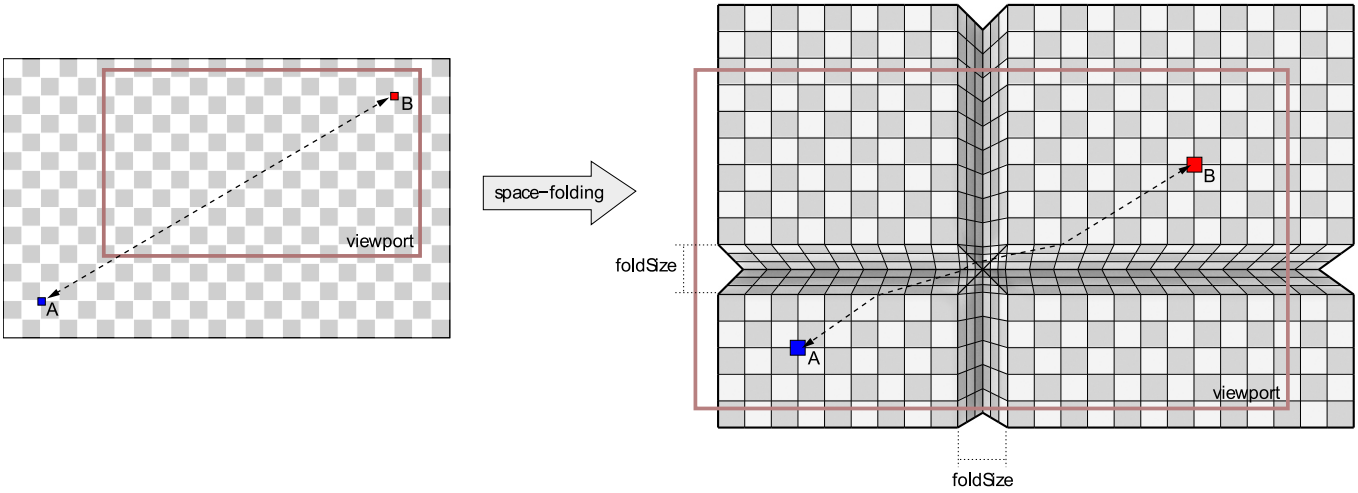


Fig. 3. Folding a 2D space with two focus points A (main) and B. Note that there are two types of folds involved here, horizontal and vertical ones.

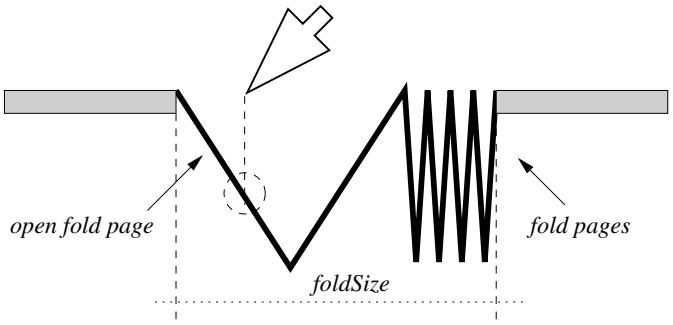


Fig. 4. Fold pages for conveying a sense of distance between focus regions. Supports flipping and defining new focus points.

Another benefit is that context awareness is improved by allocating more screen estate to each individual fold page. Pages could also show condensed context information on its one-pixel representation, akin to the compact contextual views of the City Lights [41] technique.

Hovering with the mouse over the pages flips through them like leafing through a book. Clicking on a fold adds a focus point on the designated location, and double-clicking removes all of the other focus points and creates a new primary focus point at the position. The effect is that the user stops folding space and travels to the new location.

3.4 Design Decisions

In this section we deal with some of the specific decisions involved in the design of the Mélange method.

3.4.1 Fold Geometry

The Mélange space-folding mechanism is different to most focus+context techniques in that it compresses uninteresting space as opposed to expanding the focused space. The geometry of the actual folds is an interesting design issue; to fully support the metaphor of folding paper or fabric, the space should probably be folded in a smooth curve. However, this would cause most screen estate to be afforded to the middle region of the compressed space.

Most often, the space closer to a focus region is more important than the space halfway between regions. Therefore, our default folds are sharp and angular (more like paper origami than fabric folding), similar to the Perspective Wall [23]. 3D perspective foreshortening gives a form of fisheye effect on the contents of the folds. However, our framework also supports smooth, curved folds as well as standard space compression as for rubber sheet models (although these cause loss of distance awareness).

3.4.2 1D and 2D Layout

As discussed above, Mélange can be applied to both 1D and 2D spaces. For one-dimensional visual spaces, the layout can either be horizontal

(i.e. wide, as in a horizontal timeline) or vertical (i.e. tall, as in a tree structure, see Figure 5). In both of these cases, the method for space layout is identical except for the orientation.

For 2D layout, any space intersected by both a horizontal and vertical fold becomes non-linear if the depth coordinates of the adjacent focus regions differ—see Figure 7 (for Figure 3, the depth coordinates of all four regions are identical, so this problem does not occur). Some alternatives to drawing these non-linear spaces in the intersections may be to draw compressed thumbnails of their contents instead of folds, or to discard the intersection space entirely to avoid confusing the user.

More details on layout management are discussed in Section 4.

3.4.3 Perspective Correction

Despite the fact that Mélange uses 3D graphics for its presentation, the actually technique performs all layout in 2D screen space, and then reverse-projects the points into 3D world space. This will correct for perspective to achieve an accurate visual appearance for the folds that does not introduce occlusion in the display [10]. Otherwise, the perspective projection of the 2D space deformed into 3D causes uneven distribution of screen space. Carpendale [8] calls this *folding* a region over other regions, unrelated to our use of the term.

3.4.4 Interaction Techniques

It is important to note that Mélange is merely a presentation technique for visual space, and thus does not stipulate how the user interacts with the focus points. This is a deliberate design decision to avoid coupling Mélange to a specific interaction technique. However, interacting with folded space can be fairly complex and non-intuitive, and an argument can even be made that coupling interaction and presentation might give holistic benefits not easily separated. We do present a few applications in this paper (in particular in Section 8) to give an idea of how the method might be used and interacted with.

Because of the multi-scale nature of the Mélange visual substrate, it might be useful to combine it with advanced multi-scale navigation techniques such as OrthoZoom [2] or SDAZ [19].

3.4.5 Semantic Layer

In some cases, we may want only parts of a scene to be folded, and other graphical entities to be unaffected by space transformations. Consider a node-link graph with fixed node positions but with visual edges that are designed to only show connectivity. For such a scenario, we do not particularly care if the arcs representing the edges are folded or not (Figure 8). In fact, folding can make seeing the connectivity more difficult. For the air travel scenario, if we are not interested in the exact route a flight takes from departure to arrival, not folding the edges would make distinguishing the different routes easier.

We support this situation in Mélange by introducing a semantic layer on top of the folded space layer. This layer acts as a container for adding this kind of visual items—such as annotations, labels, and visual links—that are not to be folded with the rest of the scene, but which are still anchored to points on the folded space layer.

4 Implementation

We have implemented Mélange as a Java application framework using the JOGL[1] OpenGL bindings. The framework provides a 2D scene graph populated by the application that is automatically deformed depending on the current focus points and viewpoint configuration. In this section, we will describe the implementation details of the Mélange framework, including the general system architecture, the

[1] http://jogl.dev.java.net/

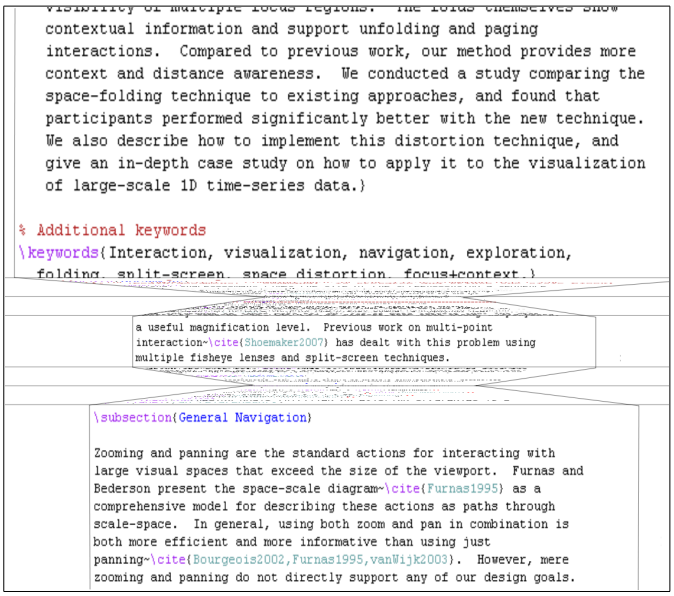


Fig. 5. Vertical layout for document visualization (e.g. comparing different parts of the document or showing multiple word occurrences).

2D layout manager, and the 3D folding functionality. Figure 5 shows an example of vertical folding functionality for a document editor.

4.1 System Architecture

The Mélange toolkit consists of three main components:

- **Platform:** The rendering platform abstraction which drives the toolkit (using OpenGL via JOGL in our implementation).
- **Substrate:** Visual space abstraction containing a dynamic 2D scene graph of the visual items to render.
- **Layout manager:** Folding mechanism that performs layout in 2D and then translates this to affine transformations associated with pieces of the scene graph entities. Maintains the active focus points in the system (including the current base focus point).

Every rendering update, the render platform calls the same sequence of commands for each active viewport: (1) it first derives the viewport dimensions and position on the visual space, then (2) calls the layout manager to compute the 2D space allocations for all focus points, and finally (3) folds the visual entities on the substrate and draws them to the screen. Rendering updates are performed whenever a viewport or any of the focus points changes, or if the scene graph is modified.

We will discuss these steps in-depth in the following subsections.

4.2 Scene Graph

The Mélange scene graph is a polymorphic Java class hierarchy consist of high-level *visual entities* representing basic graphical primitives such as rectangles, circles, line shapes, etc. Visual entities are exposed to the client programmer and allows for creating the desired visual appearance, such as a time-series visualization or a map view.

Internal to the toolkit, however, these visual entities are composed of *visual blocks*. These are graphical primitives that support splitting by an arbitrary 2D line into two or more new blocks. Splitting is used internally by the folding mechanism in the layout manager to achieve multi-focus support. There is typically a one-to-one mapping between visual entities and the visual blocks that implement them, but this facade design also allows for compound visual entities that represent

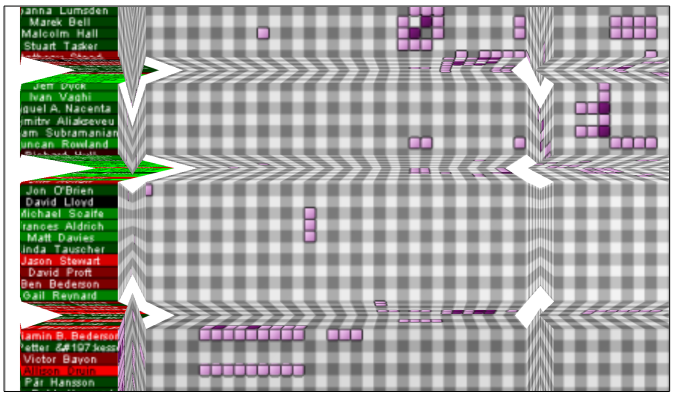


Fig. 6. 2D layout for a large graph visualized using an adjacency matrix.

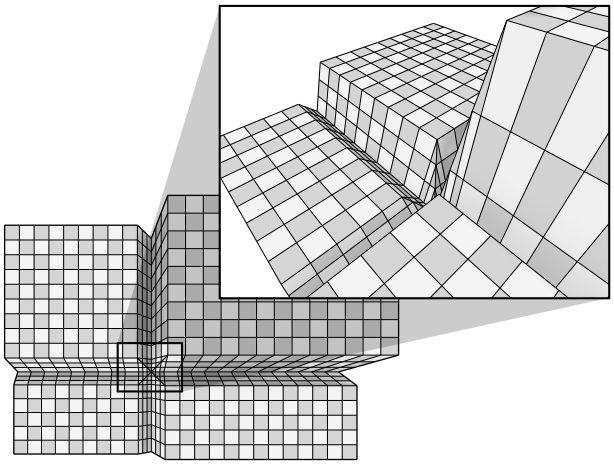


Fig. 7. Conceptual view of 2D space-folding for different zoom levels. To avoid rips in space, the distant fold lines are fixed to the same depth.

cached collections of visual blocks, such as for tesselating curved lines and surfaces or rendering text as texture-mapped bitmaps.

4.3 Viewports

Mélange supports rendering to several viewports, either organized in the same window (to, in effect, support split-screen configurations) or across different windows. Attached to each viewport is a 3D camera that is defined by a 2D position and a distance from the baseline surface of the 2D visual space. The camera also exports a projector object that translates back and forth between screen 2D coordinates and the 3D world coordinates.

Prior to rendering each frame for a particular viewport, the Mélange toolkit queries the position and extents of the viewport. In order to avoid focus points being laid out on the very border of the viewport, these extents are shrunk to add a "safe border" around the viewport, akin to safe borders in video editing. The safe viewport extents, position, and the projector is passed along to the layout manager.

4.4 Layout Management

The layout manager manages a set of focus points, which are basically positions in 3D space. The meaning of these focus points is simple: a focus point (x,y,z) addresses a point (x,y) in the 2D visual space that is requested to be visible inside the viewport at depth z from the baseline. The layout manager's mission is to make this work globally for the whole set of active focus points—in other words, to guarantee the visibility of all foci registered by the user or the application. Depending on the layout manager, it will honor one (1D vertical or horizontal) or both (2D) of the (x,y) coordinates of the focus point.

The basic layout management algorithm reduces to deriving the screen and world coordinates along a given axis of any folds that are needed to bring all active focus points into the viewport. The algorithm computes layout allocations starting from the base focus point and working outwards to the edge of the screen (in other words, left and right for horizontal layout, and up and down for vertical layout). All layout calculations are done in 2D screen space and then reverse-projected to 3D world space. This will correct all 3D geometry for perspective to avoid occlusion.

There are two cases that the layout manager needs to consider:

- **Space excess.** There is too much space (in world coordinates) between one focus point and another so that including it all will push one of the points outside the screen viewport.
- **Space shortage.** The available space (in world coordinates) between two focus points of different depths is not enough to cover their distance on the screen.

In the former case, in the presence of an excess of space, the layout manager will introduce a discontinuity in the visual space that simply cuts away a sufficient amount of intervening space so that both focus points are visible. This will be translated into a fold in the next step.

In the latter case, in the presence of space shortage, the intervening space must instead be stretched so that it covers the screen distance between focus points. Again, this is represented by a discontinuity in the visual space that is later turned into a stretch construct.

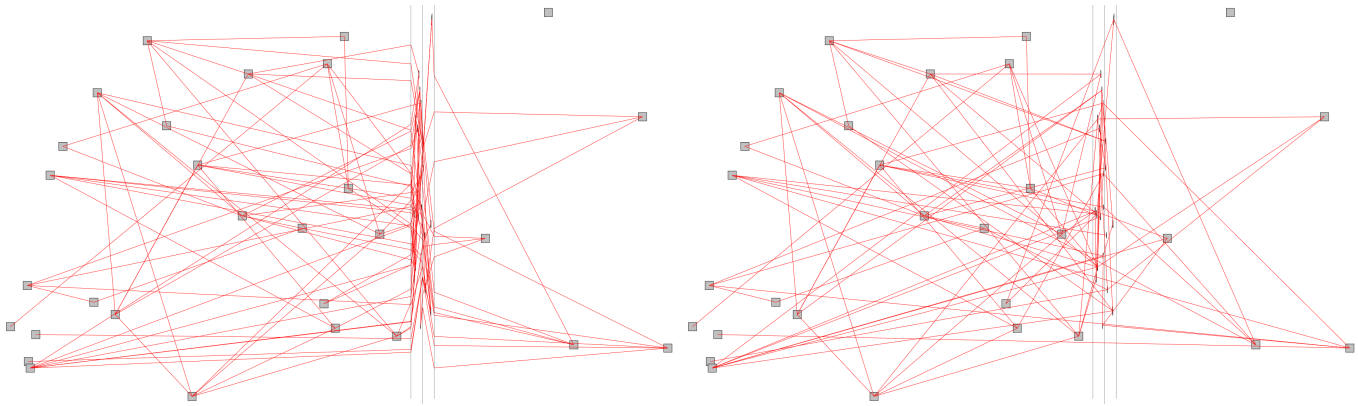
After performing layout along both directions of all involved axes, these space discontinuities are collected and sorted and then passed along to the next step in the pipeline.

4.5 Folding Space

The actual Mélange space-folding process is performed by turning the space discontinuities derived by the layout manager into a 1D or 2D space object consisting of an ordered array of spatial regions (i.e., intervals for 1D and rectangles for 2D), each with their own 3D transformation matrix. In other words, this construct describes exactly how any point on the space will be transformed when rendering to the screen. The 3D transform is computed so that all edges of the region are continuous with neighboring regions, rotating and translating the region as necessary to form a seamlessly folded presentation space.

Having built this construct, each visual entity in the scene graph of the substrate is added to the space, one at a time. Remember that visual entities consists of one or several blocks that supports splitting along arbitrary 2D lines. Folding a visual entity translates to splitting its blocks into sub-blocks that intersect a single spatial region. In other words, blocks are iteratively split along region boundaries and the resulting sub-blocks are added to the list of blocks for the particular regions they belong. The result after folding all entities in the scene is a collection of blocks sorted into the correct spatial region.

Folding 2D space is similar to 1D horizontal or vertical folding, but is conducted in two stages: first, the entities are split along the horizontal folds and organized into columns; second, each visual entity in the respective columns are split vertically into cells (Figure 6).



(a) Edges folded. (b) Edges in semantic layer.

Fig. 8. Node-link visualization of a graph with 50 nodes and 100 edges: (a) edges folded; (b) edges in semantic layer.

4.6 Rendering

When the entities have been folded, all that remains is to draw the actual scene onto the screen. This is done by simply visiting each region, setting its transform, and then drawing its contents using standard rendering commands. The transform will ensure that the output will be correctly folded and stretched. Care must be taken while splitting entities to preserve their relative depth orders, or flickering due to "z-fighting" (numerical instabilities in the depth buffer) may result.

As discussed in Section 3.4.2, the fold regions become non-linear for 2D layout of adjacent focus points with different zoom levels. Our current solution to this problem is to simply discard the space that is intersected by both a vertical and a horizontal fold. Even then, 3D perspective makes for discontinuity, skewing, and occlusion effects in the intersected fold areas (Figure 6).

Better approaches would be to either draw the non-linear space, or to abstract it by a compressed view (similar to rubber sheet [36, 24] methods). We could also fix folds to the same depth, compressing the different amounts of folded space for each region accordingly (Figure 7). However, none of these approaches have been implemented in our current version of the toolkit.

4.7 Parametrizing Folds

Fold regions can be parametrized in Mélange in order to support different kinds of fold geometries. The default fold is a single fold, translating into two spatial regions—one sloping into the depth of the screen, the other back out. However, the toolkit also has a compression parametrization that instead compresses or stretches the space as needed. In essence, this is equivalent to the rubbersheet approach, but with independently zoomed focus regions.

More complex fold types are possible. While we have not implemented this in our current toolkit, this mechanism can be used to support the fold pages discussed in Section 3.4.1, creating multiple folds for each region to communicate distance, as well as creating smooth folds by tesselating the visual space into a large number of small spatial regions, suitably transformed to form a smooth and curved surface.

4.8 Adding Semantic Layers

We support semantic layers that are not space-folded by overlaying a secondary, semantic graph on top of the standard scene graph in the Mélange implementation. Semantic nodes are similar to normal nodes, except that they have a special flag that prevents them from being folded with the rest of the scene, and causes them to be rendered last, after the folded space layer has been drawn.

Semantic nodes have one or more *anchors* which are 2D points that connect the node to a specific position on the folded space. When rendering, the Mélange framework will look up the actual 3D position of this point on the screen, and draw the semantic node at this position. Semantic nodes may have more than one anchor, in which case the node geometry will be transformed to accommodate the actual positions of the anchor points. For instance, visual edges in a node-link diagrams would have two anchor points, one for each visual node on the folded space, ensuring that the nodes remain connected regardless of how the space is transformed (Figure 8).

5 User Study

We performed a controlled experiment to evaluate whether the Mélange technique assists users in exploring large visual spaces by comparing it to single and split-screen viewports. We designed the experiment to test our design goals in the context of a matrix visualization of a large social network with MatLink [17] arcs connecting relevant nodes (connected actors) in the graph.

5.1 Participants

We recruited 12 unpaid subjects (1 female, 11 male) for our study. The participants were from 20 to 35 years of age, had normal or corrected-to-normal vision, and were screened to not be color-blind. No specific skills were required other than basic computer experience.

5.2 Apparatus

The experimental apparatus consisted of an Apple iMac Core 2 Duo 2.33 GHz workstation with 2 GBs of memory and equipped with a standard two-button mouse (with wheel) and keyboard. The 21-inch display was fixed at 1680×1050 resolution and powered by an ATI Radeon X1600 with 256 MB of video memory.

5.3 Tasks

Participants were given a source node and its neighborhood on an adjacency matrix representation of a social network, and were then asked to perform three tasks in sequence:

- T1 Find one destination node connected to the source node with the same neighborhood [G1 and G2]
- T2 Estimate the distance between the source and destination nodes (in 1:1 screen units) [G4]
- T3 Estimate the number of contextual targets between the source and destination nodes [G3]

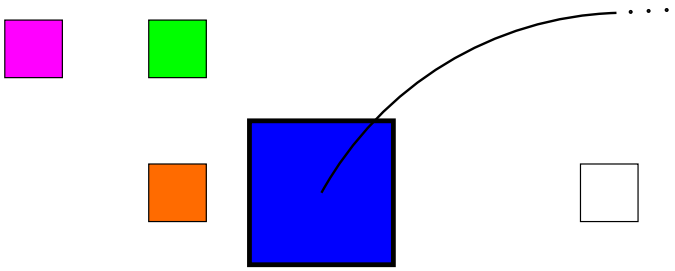


Fig. 9. Example of a source target with its four-node neighborhood.

This scenario was inspired by social network analysis where a matrix visualization of the social network is used for certain tasks. One such task is to compare the local neighborhood of two actors to find similar patterns of collaboration, but following a path can be difficult in a matrix visualization [14]. The MatLink [17] technique adds contextual visual links on the surface of the matrix to make this task easier.

Potential targets in our study were blue squares measuring 20 pixels (at 1:1 zoom level), surrounded by a neighborhood of four half-size (10 pixel) squares of different colors (Figure 9). We chose five colors for these neighborhood squares: white, magenta, orange, green, and blue (a selection that is preattentively perceptible [16]). Neighborhood nodes were placed in a 5×5 grid around the blue rectangle, and targets were placed in the same vertical position on the visual space, as if placed in different columns but the same row in a matrix visualization.

Targets were said to be identical if both the position and color of their neighborhood nodes were identical. Only one target neighborhood matched the source target, all others were distractors. Connections between the source node and the potential targets were visualized using MatLink arcs. Not all nodes on the visual space had a MatLink arc from the source node; those without were background nodes that also served as distractors, and participants were instructed to disregard them when looking for the destination target.

Contextual targets (T3) were red squares six times the size of primary targets (i.e. 120 pixels) and drawn below the primary targets. The motivation for this was that being aware of intervening context is only applicable for large-scale features such as mountain ranges or large bodies of water on a map, or communities of actors in a social network. Therefore, the intention with the contextual targets and task T3 was to measure each technique's efficiency in supporting user understanding of the visual space beyond the target nodes.

All targets—i.e., target nodes, neighborhood nodes, and contextual targets—were guaranteed to be rendered as at least a single pixel, ensuring their visibility even if the view was zoomed out or distorted.

The visual space itself was represented by a checkered gray rectangle that was 30 screens wide and one screen high. Each scenario had randomly-generated distractors. The source node was always located on the left edge of the rectangle, so the participant always had to pan right to find the target. The view was initialized to center on the source node at 1:1 zoom for every new scenario (started by T1), and was then left in its previous position for each consecutive task (T2 and T3).

To give users a frame of reference for distance, screen units were indicated on the visual space by black lines drawn on the checkered background. Figure 10 shows screenshots from our experiment application.

5.4 Predictions

P1: Mélange is as fast as single or split-screen viewport We believe that the space-folding technique will not introduce significantly slower completion times for visual search (task T1). In other words, we think that the added visual complexity and space allocations of the fold region and the additional focus point will not cause slow-downs for a user trying to locate a specific target in the space.

P2: Mélange provides more efficient context awareness None of the two techniques we compare Mélange to support contextual views explicitly, but participants are nonetheless exposed to this context when navigating over the visual space. We submit that the intervening context shown in the fold regions of the technique will cause significantly lower completion times for tasks T2 and T3.

P3: Mélange provides more accurate context awareness Analogously to P2, we believe that participants will be more accurate when answering contextual tasks (T2 and T3) with Mélange than the other two presentation techniques. Mélange provides an integrated overview of the context, whereas the other two require the user to manually pan and zoom around in the space to discover this information.

5.5 Experimental Conditions

The factors were presentation technique, off-screen distance, distractor density, and contextual target density.

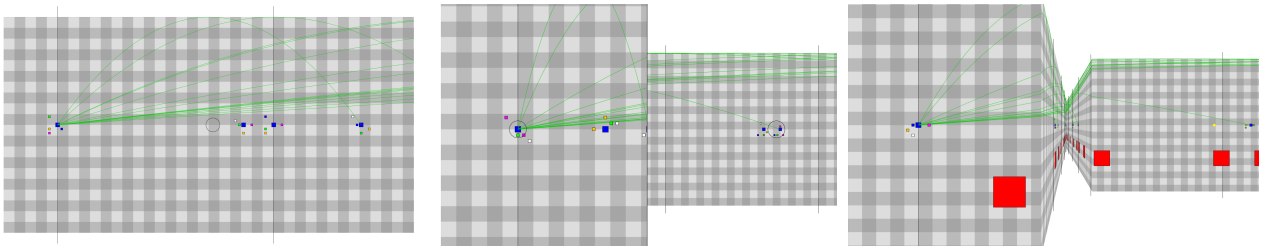
Presentation Technique The primary objective of our experiment was to study the performance of different presentations of the visual space for supporting our design goals. In addition to the Mélange technique, we included single and split-screen viewport conditions. While none of these two fulfill our design goals, they are commonly used in practice, suggesting that they are suitable competitors.

We considered comparing our technique against Accordion Drawing [24]. However, AD does not seem to support independently zoomed foci. Furthermore, Nekrasovski et al. [25] have shown that pan and zoom for a large hierarchical dataset is more efficient than navigation in AD spaces, hence our choice of competing techniques.

- *Single viewport (SV).* The standard baseline consisting of a single window showing a view of the visual space. Has no direct support for any of our stated design goals, these must be achieved through interaction.
- *Split-screen viewport (SSV).* The main viewport is split vertically into two equal-sized subwindows, each showing a different view of the visual space. In our setup, the left subwindow was fixed to always show the source node at 1:1 zoom, while the user could interact with the view of the right subwindow.
- *Mélange (M).* Our space-folding technique with the primary focus point on the source node and the secondary point controlled by the user. Moving the focus point (in the horizontal and depth dimensions) thus caused the visual space to be folded to accommodate both focus points in the viewport. Fold pages were disabled to not unfairly give a direct distance measure to the participants (i.e. only the 3D perspective foreshortening of the folds indicated distance).

All three techniques were controlled using standard zoom and pan operations (similar to standard zoomable user interface toolkits [4, 5]):

- *Panning*: Dragging the mouse while holding down the left mouse button caused horizontal movement of the focus point (i.e., pan):
  - Single viewport (SV): panning the viewport horizontally;
  - Split-screen (SSV): panning the right subwindow horizontally; and
  - Mélange (M): moving the folding focus point horizontally.
- *Zooming:* Dragging the mouse (up or down) with the right mouse button pushed—or by spinning the mouse wheel—caused depth-oriented movement of the focus point (zoom):
  - Single viewport (SV): zooming the viewport in and out;



(a) Single viewport (SV). (b) Split-screen viewport (SSV). (c) Mélange (M).

Fig. 10. Screenshots from the user study application.

- Split-screen (SSV): zooming the right subwindow in and out; and
- Mélange (M): moving the folding focus point towards or away from the camera.

As discussed in Section 3.4.4, we could easily have added a multi-scale interaction technique to control the various conditions. However, we opted to use simple zoom and pan operations to keep the interactive aspects of the experiment as simple as possible.

Off-Screen Distance We wanted to see whether performance varied with the distance to traverse on the visual space, so we tested three different distances: 4, 8, and 16 screen widths of distance (in our experimental setup, the screen width was 1680 pixels). In a matrix representation, this corresponds approximatively to networks containing 400, 800, and 1600 actors.

Distractor Density The number of false targets (i.e. distractors) between the source and destination nodes will clearly affect the time spent finding the destination node (T1). Thus, we included two different densities: *low* or *high*. This corresponded to one or two potential targets per screen (half of them background nodes with no MatLink arcs to them).

Contextual Target Density We studied two levels of density for the contextual targets between the source and destination nodes: *few* (less than or equal to five) or *many* (more than five).

5.6 Experimental Design

We used a $3 \times 3 \times 2 \times 2$ within-subjects factorial design. In summary, the factors (described above) were:

- Presentation technique: single (SV), split (SSV), or Mélange (M)
- Off-screen distance: 4, 8, or 16 screens
- Distractor density: 1 or 2 per screen (average)
- Contextual target density: few (≤ 5) or many (> 5)

The order of the techniques was counterbalanced (two participants assigned to each order). Participants were asked to complete 3 blocks—one per technique—of 24 trials (3 distances $\times$ 2 distractor densities $\times$ 2 contextual target densities $\times$ 2 trials) in randomized order. With 12 participants, the study software collected 864 trials in total.

5.7 Procedure

Participants were introduced to the study and randomly assigned to one of the six order groups for the presentation technique. They then performed three blocks of trials, one per technique, in succession. Before each block, the test administrator explained how to use the technique and then let the participant practice on six training trials. Participants were not allowed to proceed past each training trial without answering correctly to all three tasks.

Task	Factors	F	p
T1	Distance	38.740	**
	Distractors	55.155	**
T2	Technique	8.695	*
	Distance	6.560	*
	Technique * Distance	6.658	**
	Distance * Distractors * Context	4.216	*
T3	Distance * Context	5.335	*
	Technique * Distance * Context	2.660	*

* = $p \leq 0.05$, ** = $p \leq 0.001$.

Table 2. Significant effects of completion time on the factors.

Each trial consisted of performing the three tasks T1 to T3 in sequence. A screen with instructions was given prior to each task, and the participant proceeded to the task by clicking a button or pressing the space bar. Task T1 ended when the participant clicked the right target (which then turned from blue to yellow); for the other tasks, the participant pressed the space bar to end the task. After task T2 and T3, participants were presented with a multiple-choice question asking about their answer to the task.

Participants were instructed to work as quickly as possible. For every trial, the software silently collected the time and correctness measures for the three tasks (only time for T1). Participants were instructed to pause between each block to avoid fatigue affecting the results. At the end of the test, they were given a preference questionnaire to complete.

6 Results

6.1 Completion Time

Table 2 summarizes the main effects for time. Figure 11 shows mean time to completion for all three tasks.

For task T1, the average completion time was 18.05 (s.d. 1.42) seconds for SV, 16.98 (s.d. 0.85) seconds for SSV, and 19.18 (s.d. 0.99) seconds for M (SSV $<$ M $<$ SV). An analysis of variance (ANOVA) showed no significant main effect of Presentation technique.

For task T2, the average time was 4.13 (s.d. 0.64) seconds for SV, 4.02 (s.d. 0.43) seconds for SSV, and 2.74 (s.d. 0.35) seconds for M (M $<$ SSV $<$ SV). ANOVA yielded a significant main effect for Presentation technique ($F_{2,22} = 9.203, p = .001$).

For T3, the average time was 1.72 (s.d. 0.57) seconds for SV, 1.90 (s.d. 0.50) seconds for SSV, and 1.64 (s.d. 0.19) seconds for M (SV $<$ M $<$ SSV). ANOVA yielded no significant main effect for Presentation technique.

6.2 Correctness

For task T2, the average correctness was 0.986 (s.d. 0.007) for SV, 0.948 (s.d. 0.013) for SSV, and 0.983 (s.d. 0.008) for M (SV $>$ M $>$

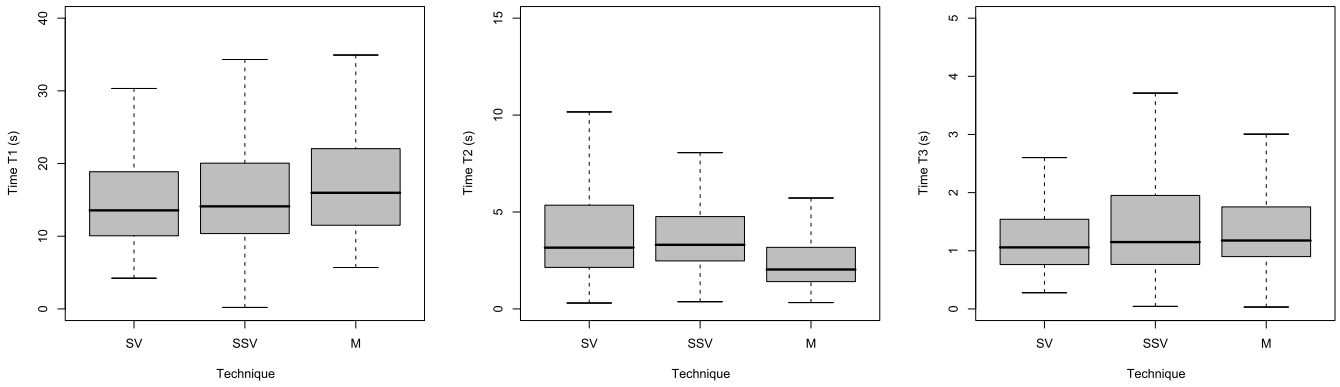


Fig. 11. Average completion times for presentation technique across T1, T2, and T3.

SSV). This is a significant difference (Friedman test, $p = .008$). A Wilcoxon test for paired comparison shows that M and SV have higher correctness than SSV (M vs SSV: $p < .025$, SV vs SSV: $p < .012$). Figure 12 shows the mean correctness for T2.

For task T3, the mean correctness was 0.983 (s.d. 0.008) for single viewport, 0.965 (s.d. 0.011) for split-screen, and 0.983 (s.d. 0.008) for Mélange—not a significant difference (Friedman test, $p = .189$).

6.3 Subjective Preference

When asked about their preference on the presentation technique, 5 out of 12 participants ranked the Mélange technique first (5 for split-screen and 2 for single viewport). Comments from the participants were favorable for our new technique, particularly for contextual tasks.

7 Discussion

Summarizing the results, our user study yields the following findings:

- Our experiment shows no significant differences between the three techniques for visual search (T1) so we cannot conclude about our prediction P1. With 12 participants, the techniques seemed comparable in performance.
- Mélange is significantly faster for the contextual task T2 than both single and split-screen viewport, confirming prediction P2. The difference is almost one-third of the completion time for the competing techniques.
- Mélange promoted significantly better correctness than split-screen viewport. This partially confirms prediction P3. There was no difference for Mélange in comparison to single viewport, but this may be due to single viewport simply not supporting quick contextual assessment and thus required careful (and time-costly, as the results show) visual search.

In the following sections, we try to explain and generalize these results, and see how our work can be used in practice.

7.1 Explaining the Results

These results confirm that the Mélange space-folding technique provides extra benefit beyond the standard split-screen method. More specifically, the results show that providing an awareness of intervening context and distance between focus points helps for contextual tasks, while clearly not consuming too much screen space or cognitive effort to cause poorer performance than split-screen viewports.

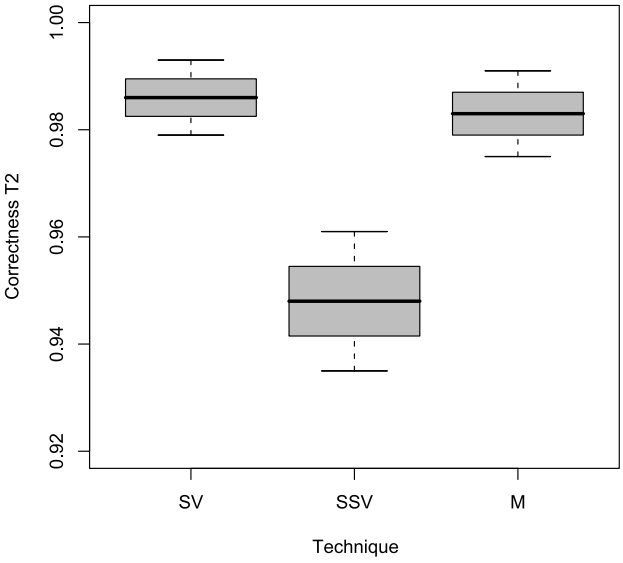


Fig. 12. Correctness for presentation technique for T2.

Looking at the completion times for task T1, we note that the difference between single-focus (single viewport) and the two double-focus (split-screen and Mélange) presentation techniques is small (within 10% of the average search time). The reason for this is that T1 is a relatively simple visual search task where the target appearance can be memorized (as reported in post-test interviews by several participants), so two foci are not strictly necessary for this task and in fact may have slowed down users during the visual search.

We designed the study this way to avoid punishing the participants with very long completion times—instead, the objective of task T1 (rather than strictly confirming G1 and G2) is to show that space-folding does not introduce slow-downs in navigation compared to single or split-screen viewports (prediction P1). Because our results are not significant, we cannot make a claim either way about this hypothesis, but we nevertheless believe that the results support it.

We found no significant difference in completion time for the T3 task, so our prediction P2 only holds for contextual task T2. However, we

observed that participants in the user study tended to solve both T2 and T3 simultaneously during the T2 time. This was possible because distance indicators and contextual targets were visible for both tasks. If we combine the completion times for both tasks, the average time was 5.74 seconds for SV, 5.79 seconds for SSV, and 4.17 seconds for M (M < SV < SSV). Removing outliers, this is a significant difference ($F_{2,22} = 4.289, p = .027$).

While Mélange was significantly more correct than split-screen, there was no difference in comparison to single viewport. We believe this is due to single viewport (as well as split-screen) simply not supporting quick context assessment. Instead, participants were forced to trade time for accuracy and perform a careful visual search (for both SV and SSV) to be able to get the same accuracy. This is also consistent with our timing results. In contrast, with Mélange, users were able to easily retrieve the contextual information with little extra effort.

7.2 Generalizing the Results

Our results show that the Mélange technique fulfills most of our predictions for the chosen scenario and tasks. The question is naturally whether these results generalize to the whole class of large visual spaces discussed in the introduction.

The answer to this question is two-fold: We believe that the tasks and the scenario used in the study are realistic enough to be ecologically valid, yet general enough to allow us to extend the results to other domains. For the first point, the tasks selected are based on typical user tasks for network analysis [22]. For the second, the study scenario is sufficiently abstract so that there is nothing in the tasks or the scenario that limits the results. Due to these reasons, it is reasonable to conjecture that the tasks can form building blocks for higher-level user tasks, such as visual exploration. In fact, for the world map example (depicted in Figure 1(b)), contextual tasks may become even easier due to the inherent multi-scale properties of a map (i.e. large-scale features like ocean, land, and mountains are visible even from long distances and under great distortion).

One specific threat to generalizing the results is that we only tested one-dimensional navigation (horizontal) in one direction (left to right). Two-dimensional tasks may exhibit differences depending on the relative positions of the foci, and laterality may also have an impact. Studying these effects remains to be investigated in the future.

For larger distances (more than the 16 screens tested in our study), the performance may degrade since the folds become very small and dense. This would happen when navigating a DNA sequence, for example. Supporting this situation is again left for future work.

Finally, it is worth reiterating that while rubber sheet-based approaches [24, 36] may appear to be better comparisons to Mélange than the split-screen and pan-and-zoom conditions tested in our experiment, the findings of Nekrasovski et al. [25] indicate that simple pan-and-zoom outperforms Accordion Drawing, even for contextual tasks and regardless of the presence of an overview map. We can easily configure the Mélange framework to emulate AD by disabling zooming of focus points and forcing compression instead of folding between focus regions. However, relying on Nekrasovki's result, we opted not to include this condition to keep the experiment manageable in size.

7.3 Multi-Focus Interaction in Practice

One important issue with all multiple-foci techniques, including split-screen and space-folding as well as overview windows, is that they divide the user's attention between several different viewports and consume valuable screen estate. Even for a focus+context technique like Mélange, there is a non-trivial cognitive effort associated with comparing the different focus regions. As for screen space, users typically interact with only one area of the visual space at a time, so multiple-foci techniques reduce the amount of screen space available for this interaction. Mélange is slightly worse than split-screen due to the fold regions also consuming screen space. Having just a single viewport sidesteps both of these concerns. However, this loss of screen space is balanced by improved context awareness.

As has been shown in this paper, split-screen is perhaps the primary competitor to space-folding. One of its major advantages is its simplicity, both for interaction and implementation. Mélange is unquestionably more complex in both aspects, but we believe that its advantages outweigh this fact. Not only does space-folding better show contextual information, as has been proven in this paper, but it also integrates several foci into the same continuous view, and directly gives the relative positioning of the foci. By the same token, split-screen viewports are fully independent of each other, so they give no intrinsic indication of what part of the space they are showing in relation to the others. In fact, both subviewports may be showing the same target, causing the user to mistake the source node for the destination node, as happened to one of our study participants.

By the same token, there are several other alternatives to support multi-focus interaction beyond Mélange. Some alternatives include augmenting existing techniques, such as split-screen and rubbersheet methods with distance (and, in some cases, direction) annotations on the visual space. We could also envision restricting ourselves to 1D data, and then use the second dimension for laying out a hierarchy of interactive focus regions. This work is far from the last word on this topic, and we anticipate continuing this investigation in the future.

We can also anticipate many other applications for multi-focus interaction and Mélange beyond those discussed in this paper. Figure 13 shows an example of a video editing timeline—essentially a 1D visual structure—being folded using our technique. This may be useful for an editor who is synchronizing shots in a video, or looking to perform color correction between different clips on the timeline.

Other potential applications could include finding sections in a large text document using a word search and matching words in the surrounding paragraphs, looking for patterns and trends of geospatial data overlaid on 2D maps that occur in several locations, and even deforming user interface components in applications containing complex menus and toolbars.

8 Case Study: Time-Series Data Visualization

In order to test the versatility of the Mélange technique, we used it to visualize time-series data [32]. Temporal data is ubiquitous in society today, and the main problem that we often face is that the time series tend to be prohibitively long, containing days, if not weeks, of high-resolution data. Analysts studying temporal data often need multi-focus interaction support to be able to correlate and compare different parts of a time sequence to look for recurring patterns as well as events out of the ordinary in the data.

8.1 MarkerClock

Some of our work concerns computer-supported design and mechanisms to support elderly people to age in place, and one of the central themes here is the concept of *PeerCare* [33]: shared awareness where elders form a peer support group to help and look out for each other. In particular, after considerable field work and prototype deployments, we identified a need for an unobtrusive technology probe [18] for sharing awareness cues between elders and their social networks.

In response to this need, we developed the MarkerClock [34] concept—a clock-based communication appliance [26] for distributing privacy-preserving social awareness cues. MarkerClock is modeled after a hand-based analog clock and is designed to be hung on a wall. The initial design was inspired by the idea of providing an ambient communication of people's environmental cues.

To support awareness of routines and rhythms, MarkerClock is able to capture cues in the users' environment (such as light intensity or mo-

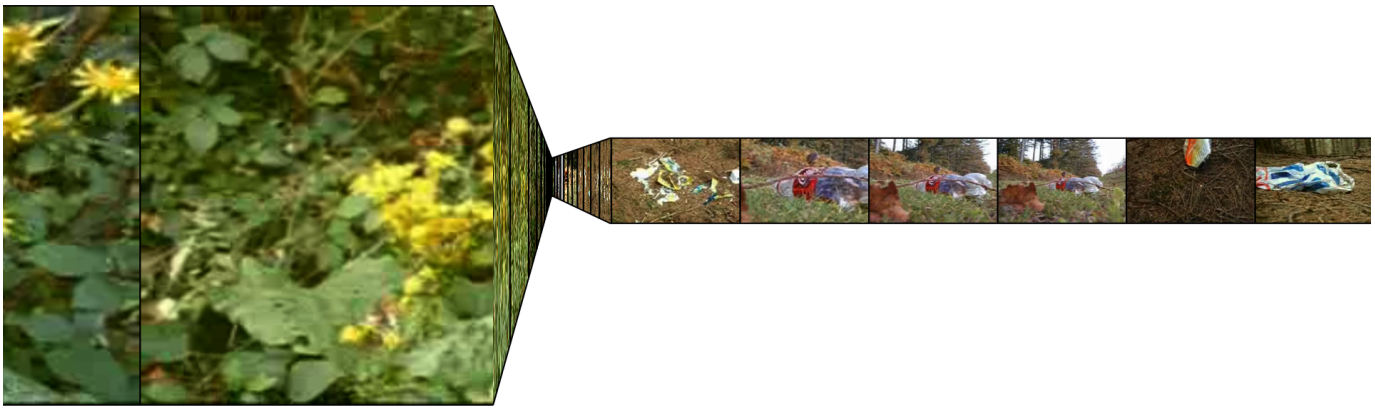


Fig. 13. Folding a 1D video editing timeline using the Mélange technique.

tion) and display the information and its history on its clock face. The clock is already a ubiquitous device in home environments, and provides an explicit, well-known mapping between angular position and time. MarkerClock makes use of that existing knowledge by placing information around the clock at the angular position corresponding to the time of the represented information.

A MarkerClock is designed to work in connection with other clocks over the network, and shows a time trace for the connected clocks on its own clock face. It captures and displays activity in its surroundings by measuring motion on a video feed provided by a webcam attached to the clock. In addition, users can drop symbols on the current time of the trace as markers to communicate with other connected clocks. See Figure 14 for an illustration of the MarkerClock design.

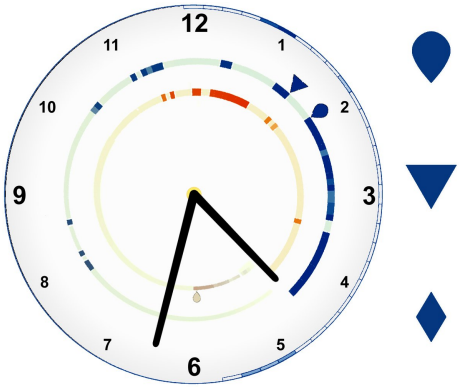


Fig. 14. MarkerClock design, including the available marker symbols. Trace information about the own clock (blue) and a remote clock (red) is shown on the interior of the clock face.

8.2 Capturing and Visualizing Temporal Data

We deployed the MarkerClock technology probe between two pairs of homes. Each deployment lasted about four weeks, during which temporal data about motion activity and use of markers was captured. Over the time of the deployments, the clock was functional 95% of the time. In total, the deployments generated 35,619 individual events, spread across just above 56 days (two deployments of four weeks).

In order to explore this data, we developed a program for aggregating the data according to time. Using this tool, we sorted the dataset into slices of 30 minutes and then fed the results as input to a simple Microsoft Excel visualization (Figure 15). This visualization and the summarized data associated with it provided an overview of the overall data but did not really allow an exploratory analysis of the fine-grained data, such as what type of marker was used and in what order. This type of exploration required the view of the details of each marker, especially since we were interested in observing if people created marker codes, i.e. combinations of markers to create a new symbol. To be able to detect such combinations, we needed to view the data at a resolution similar to the one available on the clock.

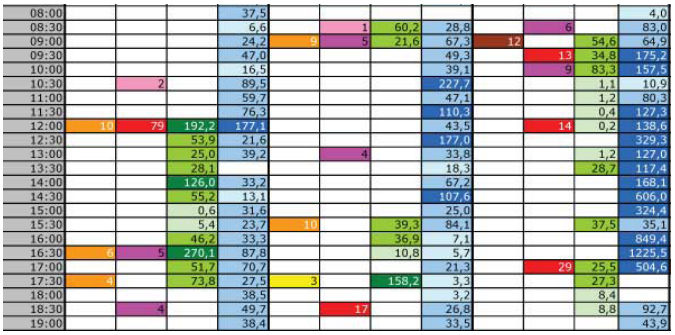


Fig. 15. Aggregated data slices visualized in Microsoft Excel.

As a result, we extended this work by creating a visualization of our quantitative data which allowed us to represent all the data without losing details. This visualization was designed to render the data as a large visual space showing the data on a timeline spanning over the duration of the whole data collection, much similar to the MarkerClock's own display (Figure 16). This representation meant that the visual space representing just a day of data was extremely large. For instance, with a time mapping of 20 seconds per pixel, we get 4,320 pixels per day, or 120,960 pixels for four weeks of deployment.

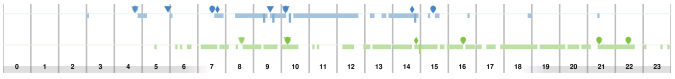


Fig. 16. Activity-based visual representation of a day of data.

8.3 Using Mélange for Time-Series Visualization

As described earlier, the analysis of the data from the two longitudinal deployments required looking both at the markers at a high level of detail, as well as looking at the details of the motion patterns at lower levels of details. Furthermore, the comparison of marker usage also required keeping an awareness of how separate details portions of the data showing markers were respectively situated (e.g. time of the day, how many days apart, etc) [32].

In other words, the analysis of the MarkerClock data turned out to be a prime example of a multi-focus interaction task, and we em-

ployed Mélange to solve the problem. Figure 17 is a screenshot of the visualization system showing two time series in the same display corresponding to two separate mornings of the same deployment for marker use comparison. We developed this application in Java using the Mélange toolkit simply by creating a long series of rectangles in 2D space and texture-mapping them with the images representing the output from the visualization tool. The toolkit handles all of the mechanics of space-folding, allowing the application designer to focus on the details of the visualization and not the mode of presentation.

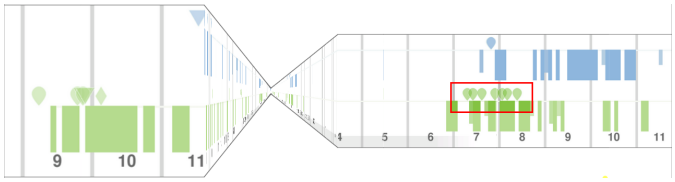


Fig. 17. Exploration of MarkerClock data using Mélange. The highlighted portion of the data shows one participant's use of markers over two hours to send a happy 70th birthday message to another participant.

The use of Mélange allowed us to support the main requirements for analyzing the data. However, to ease the appreciation of distance in the fold, we added a gradient bar in the visualization showing daylight periods during the deployment. Furthermore, when distance between focus points increased beyond four days, assessing the distance between the focus points became difficult. We believe that the use of folds, with fold size customized to fit a particular unit of analysis (such as 4,320 pixels for a day), could support better distance awareness.

This being an essentially exploratory task, Mélange allowed us to browse the data without having prior conceptions about its content. This is one of the main benefits of visual exploration [21]. However, the simple interactions for navigating the visualization and creating folds sometimes became difficult to use. In particular, more intuitive and powerful interactions are needed for creating, moving (including zooming), and deleting focus points, as well as sticking focus points to fixed "anchors" in 3D space.

Furthermore, the implementation of Mélange that was used for this case study only supported two focus regions.[2] Supporting multiple foci would have benefited the exploration by allowing side-by-side comparision of more than two marker regions at any point in time.

By reducing the fold size to zero, Mélange became comparable to using split-screen, with the exception that fold positions explicitly matched the data direction. These split-screen views were useful for tasks where context was less important than detail and the the user felt better off to use the whole screen for displaying focus points. Building on this idea, a useful feature would be a button to switch between split and folded views depending on the current exploration task.

9 Conclusion and Future Work

We have introduced Mélange, a space-distortion technique that folds 2D space into 3D to guarantee visibility of multiple focus points. The technique supports our definition of large visual space exploration by showing the distance and the intervening context between the foci. This is also what distinguishes it from existing distortion-based techniques. We have presented results from a controlled experiment that confirms the technique's usefulness in this regard.

Space-folding is our first step to supporting exploration of large visual spaces. We have so far focused mostly on the visualization aspects for the Mélange technique and less on the interaction. In the future, we anticipate designing more sophisticated interaction techniques to directly support the exploration design goals. We are also interested in continuing to study its use for visualization of social networks.

[2]Note that the current implementation presented in this paper supports an unlimited number of folds.

Acknowledgments

This work was partially funded by the French ANR Autograph project and the joint Microsoft Research/INRIA ReActivity project. The authors would like to thank the anonymous participants in the user study. Also, thank you to the ACM CHI 2008 reviewers and the IEEE TVCG reviewers, as well as others who have offered feedback on this work since it was originally conceived. Thanks to Robert Karlsson for greatly improving the figures in this paper.

The inspiration for the technique in this paper comes from Frank Herbert's classic science-fiction novel *Dune* from 1965, where interstellar space travel is performed through a process known as "folding space". Here, specially trained navigators with prescient abilities utilize the influences of a special spice known as *Mélange* to traverse fold-space and jump from one end of the galaxy to another.

> *The most precious substance in the universe is the spice Mélange. The spice extends life. The spice expands consciousness. The spice is vital to space travel. The Spacing Guild and its navigators, who the spice has mutated over 4000 years, use the orange spice gas, which gives them the ability to fold space. That is, travel to any part of the universe without moving.*
>
> — Princess Irulan's introduction to the movie adaptation of *Dune* (David Lynch, 1984).

References

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

[2] C. Appert and J.-D. Fekete. OrthoZoom scroller: 1D multi-scale navigation. In *Proceedings of the ACM CHI 2006 Conference on Human Factors in Computing Systems*, pages 21–30, 2006.

[3] P. Baudisch and R. Rosenholtz. Halo: a technique for visualizing off-screen objects. In *Proceedings of the ACM CHI 2003 Conference on Human Factors in Computing Systems*, pages 481–488, 2003.

[4] B. B. Bederson, J. D. Hollan, K. Perlin, J. Meyer, D. Bacon, and G. W. Furnas. Pad++: A zoomable graphical sketchpad for exploring alternate interface physics. *Journal of Visual Languages and Computing*, 7:3–31, 1996.

[5] B. B. Bederson, J. Meyer, and L. Good. Jazz: An extensible zoomable user interface graphics toolkit in Java. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, pages 171–180, 2000.

[6] F. Bourgeois and Y. Guiard. Multiscale pointing: facilitating pan-zoom coordination. In *Extended Abstracts of the ACM CHI 2002 Conference on Human Factors in Computing Systems*, pages 758–759, 2002.

[7] S. K. Card and D. Nation. Degree-of-interest trees: A component of an attention-reactive user interface. In *Proceedings of the ACM Conference on Advanced Visual Interfaces*, pages 231–245, 2002.

[8] M. S. T. Carpendale and C. Montagnese. A framework for unifying presentation space. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, pages 61–70, 2001.

[9] N. Elmqvist, N. Henry, Y. Riche, and J.-D. Fekete. Mélange: Space folding for multi-focus interaction. In *Proceedings of the ACM CHI 2008 Conference on Human Factors in Computing Systems*, pages 1333–1342, 2008.

[10] N. Elmqvist and P. Tsigas. A taxonomy of 3D occlusion management for visualization. *IEEE Transactions on Visualization and Computer Graphics*, 14:1095–1109, 2008.

[11] G. W. Furnas. Generalized fisheye views. In *Proceedings of the ACM CHI'86 Conference on Human Factors in Computer Systems*, pages 16–23, 1986.

[12] G. W. Furnas. A fisheye follow-up: further reflections on focus + context. In *Proceedings of the ACM CHI 2006 Conference on Human Factors in Computing Systems*, pages 999–1008, 2006.

[13] G. W. Furnas and B. B. Bederson. Space-scale diagrams: Understanding multiscale interfaces. In *Proceedings of the ACM CHI'95 Conference on Human Factors in Computing Systems*, pages 234–241, 1995.
[14] M. Ghoniem, J.-D. Fekete, and P. Castagliola. On the readability of graphs using node-link and matrix-based representations: a controlled experiment and statistical analysis. *Information Visualization*, 4(2):114–135, 2005.
[15] S. G. Gustafson and P. P. Irani. Comparing visualizations for tracking off-screen moving targets. In *Extended Abstracts of the ACM CHI 2007 Conference on Human Factors in Computing Systems*, pages 2399–2404, 2007.
[16] C. G. Healey. Choosing effective colours for data visualization. In *Proceedings of the IEEE Conference on Visualization*, pages 263–270, 1996.
[17] N. Henry and J.-D. Fekete. MatLink: Enhanced matrix visualization for analyzing social networks. In *Human-Computer Interaction — INTERACT 2007*, volume 4663 of *LNCS*, pages 288–302, 2007.
[18] H. Hutchinson, W. Mackay, B. Westerlund, B. B. Bederson, A. Druin, C. Plaisant, M. Beaudouin-Lafon, S. Conversy, H. Evans, H. Hansen, N. Roussel, and B. Eiderback. Technology probes: inspiring design for and with families. In *Proceedings of the ACM CHI 2003 Conference on Human Factors in Computing Systems*, pages 17–24, 2003.
[19] T. Igarashi and K. Hinckley. Speed-dependent automatic zooming for browsing large documents. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, pages 139–148, 2000.
[20] P. Irani, C. Gutwin, and X. D. Yang. Improving selection of off-screen targets with hopping. In *Proceedings of ACM CHI 2006 Conference on Human Factors in Computing Systems*, pages 299–308, 2006.
[21] D. A. Keim. Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):1–8, 2002.
[22] B. Lee, C. Plaisant, C. S. Parr, J.-D. Fekete, and N. Henry. Task taxonomy for graph visualization. In *Proceedings of BEyond time and errors: novel evaLuation methods for Information Visualization (BELIV'06)*, pages 82–86, 2006.
[23] J. D. Mackinlay, G. G. Robertson, and S. K. Card. The Perspective Wall: Detail and context smoothly integrated. In *Proceedings of the ACM CHI'91 Conference on Human Factors in Computing Systems*, pages 173–179, 1991.
[24] T. Munzner, F. Guimbretière, S. Tasiran, L. Zhang, and Y. Zhou. TreeJuxtaposer: scalable tree comparison using focus+context with guaranteed visibility. In *Proceedings of ACM SIGGRAPH 2003*, pages 453–462, 2003.
[25] 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 *Proceedings of the ACM CHI 2006 Conference on Human Factors in Computing Systems*, pages 11–20, 2006.
[26] B. V. Niman, D. V. Thanh, J. Herstad, and H. Hüttenrauch. Transparent communication appliances. In *Proceedings of HCI International*, pages 18–22, 1999.
[27] K. Perlin and D. Fox. Pad: An alternative approach to the computer interface. In *Proceedings of Computer Graphics (SIGGRAPH 93)*, pages 57–64, 1993.
[28] C. Plaisant, D. Carr, and B. Shneiderman. Image browsers: Taxonomy and guidelines for developers. *IEEE Software*, 12(2):21–32, Mar. 1995.
[29] C. Plaisant, J. Grosjean, and B. B. Bederson. SpaceTree: Supporting exploration in large node link tree, design evolution and empirical evaluation. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 57–64, 2002.
[30] C. Plaisant, B. Milash, A. Rose, S. Widoff, and B. Shneiderman. LifeLines: Visualizing personal histories. In *Proceedings of the ACM CHI'96 Conference on Human Factors in Computing Systems*, pages 221–227, 1996.
[31] R. Rao and S. K. Card. The Table Lens: Merging graphical and symbolic representations in an interactive focus+context visualization for tabular information. In *Proceedings of the ACM CHI'94 Conference on Human Factors in Computing Systems*, pages 318–322, 1994.
[32] Y. Riche. *Designing Communication Appliances to Support Aging in Place*. PhD thesis, LRI, Université Paris-Sud, 2008.
[33] Y. Riche. Peercare, routines and rhythms for better aging in place. *Journal of Collaborative Computing*, 2009. to appear.
[34] Y. Riche and W. Mackay. markerClock: A communicating augmented clock for elderly. In *Proceedings of INTERACT*, pages 408–411, 2007.
[35] G. G. Robertson and J. D. Mackinlay. The Document Lens. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, pages 101–108, 1993.
[36] M. Sarkar, S. S. Snibbe, O. J. Tversky, and S. P. Reiss. Stretching the rubber sheet: A metaphor for visualizing large layouts on small screens. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, pages 81–91, 1993.
[37] G. Shoemaker and C. Gutwin. Supporting multi-point interaction in visual workspaces. In *Proceedings of the ACM CHI 2007 Conference on Human Factors in Computing Systems*, pages 999–1008, 2007.
[38] J. Slack, K. Hildebrand, and T. Munzner. PRISAD: A partitioned rendering infrastructure for scalable accordion drawing (extended version). *Information Visualization*, 5(2):137–151, 2006.
[39] J. J. van Wijk and W. A. A. Nuij. Smooth and efficient zooming and panning. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 15–22, 2003.
[40] A. Zanella, M. S. T. Carpendale, and M. Rounding. On the effects of viewing cues in comprehending distortions. In *Proceedings of the ACM NordiCHI Conference on Human-Computer Interaction*, pages 119–128, 2002.
[41] P. T. Zellweger, J. D. Mackinlay, L. Good, M. Stefik, and P. Baudisch. City lights: contextual views in minimal space. In *Extended Abstracts of the ACM CHI 2003 Conference on Human Factors in Computing Systems*, pages 838–839, 2003.



Niklas Elmqvist is an assistant professor in the School of Electrical and Computer Engineering at Purdue University in West Lafayette, IN, USA. Having joined Purdue in Fall 2008, he was previously a Microsoft Research postdoctoral researcher in the AVIZ team of INRIA Saclay - Île-de-France located at Université Paris-Sud in Paris, France. He received his Ph.D. in December 2006 from the Department of Computer Science and Engineering at Chalmers University of Technology in Göteborg, Sweden. His research specialization is in information visualization, human-computer interaction, and visual analytics. He is a member of the IEEE and the IEEE Computer Society.



Yann Riche is an independent researcher in Seattle, WA, USA, recently graduated from his Ph.D. in human-computer interaction at Université Paris-Sud. His research interests lie in computer-mediated communication, design for the home, and computer-supported collaborative work. During his Ph.D., he explored the design and evaluation of communication devices for supporting aging in place. His other projects included the exploration of physical collaboration in design, the use of technology to support teaching in high schools, and the design of tools to support data capture and analysis in the field.



Nathalie Henry is a researcher in the VIBE group at Microsoft Re-

search. Her interests lie in social networks, network visualization, information visualization, and general human-computer interaction. After a master's in computer graphics, Nathalie earned a cotutelle Ph.D., sharing her time working with Jean-Daniel Fekete and the AVIZ team at INRIA and Université Paris-Sud in France, and Peter Eades and graph drawing researchers at the University of Sydney in Australia. Her Ph.D. research focused on the visualization of social networks using matrix-based representations.



Jean-Daniel Fekete is a Senior Research Scientist (DR2) at INRIA Saclay - Île-de-France, one of the leading French national research centers, in Orsay, south of Paris. He leads the AVIZ team since 2007, which focuses on data analysis and visualization research. The AVIZ team is located in and collaborates with the Computer Science Department (LRI) at Université Paris-Sud. AVIZ studies analysis and visualization of large datasets, combining machine learning approaches with information visualization and multiscale interaction techniques to help analysts explore and understand massive data. Jean-Daniel's research topics include network visualization, evaluation of information visualization systems, and toolkits for user interfaces and information visualization. His research is applied in several fields such as biology, history, sociology, digital libraries, and business intelligence. He is a member of the IEEE.