Buoys, break lines, and unique backgrounds: techniques for non-disruptive bidirectional spatial links

Tuomas J. Lukka, Janne V. Kujala,
Matti Katila and Benja Fallenstein
Hyperstructure Group
Agora Center, University of Jyväskylä
P.O. Box 35, FIN 40014
lukka@iki.fi, jvk@iki.fi, mudyc@iki.fi,
b.fallenstein@gmx.de

ABSTRACT

We present new user interface techniques to facilitate non-disruptive user interfaces to hypertext, made possible by modern graphics accelerators. *Buoys* are objects floating in the margin, visibly connected to an anchor in a document. *Break lines* make fragments look like torn-off pieces of the link targets. *Unique backgrounds* for documents allow users to recognize a fragment's source. Using a *nadir* for orienting the buoys makes them visually distinct. The techniques are especially effective when used in combination.

Together, these techniques allow the target anchors of all currently visible links to be shown. Upon traversing a link, the view can fluidly animate to the target, while the originating document remains visible, moving into the margin.

As an example, we show screenshots of a prototype for allowing structured user annotations between PDF files.

Categories and Subject Descriptors

H.5.2 [**Information Interfaces and Presentation (e.g., HCI)**]: User Interfaces—*Graphical user interfaces (GUI), Screen design (e.g., text, graphics, color)*; H.5.4 [**Information Interfaces and Presentation (e.g., HCI)**]: User Interfaces, Hypertext/Hypermedia—*Navigation, User issues*; I.3.6 [**Computer graphics**]: Methodology and Techniques—*Interaction techniques*

General Terms

Design, Human Factors

Keywords

Focus+context, Xanalogical, Buoy

1. INTRODUCTION

In several hypertext systems today, following a link means a *disruptive* change in the user's workflow, replacing the current context (e.g., a Web page) with an entirely different one. This is partly caused by the dominant graphical user interface paradigm (developed in the 70s at Xerox PARC; see, e.g., [6]) in which *pages* are shown in overlapping, rectangular, unconnected viewports (windows). In this paradigm, following a link can only create a new window or replace the contents of the current window. In our opinion, this is one of the root causes of hypertext disorientation [4] [7, pp. 38-40].

Improving user orientation through user interface improvements has recently received much attention. One early approach is to display information about the destinations of links. Browsers' status lines or tooltips show the URL or the HTML link title of the link that the mouse is currently over. Fluid links [21] take the concept further by allowing the user to see gradually more and more target context inserted into the current document before they follow a specific link. Hypercept [8] provides a cue of local structure by animating the transition from the current document to a linked document in different ways depending on the structural relationship exemplified by the link.

The Pad++ browser [2] records history of the visited pages as a tree with nodes showing complete pages. The focused page is shown at a larger scale and the user can pan and zoom the large virtual view with all layout changes fluidly animated.

Nelson's transpointing windows [12] show connections cross-cutting the view hierarchy between relevant parts of documents. This form of annotation is often used on images: a label is placed on the margin and a line is drawn to the relevant point in the image, see Fig. 1.

Another approach is to treat the local hypertext structure as a graph, and visualize it as such. Focus+context views of the web, as proposed by Mukherjea and Hara [9] and Munzner and Burchard [10], provide overview diagrams of the linking strucure of web pages with important nodes emphasized.

Free form 'digital ink' annotation (e.g., XLibris [18], iMarkup

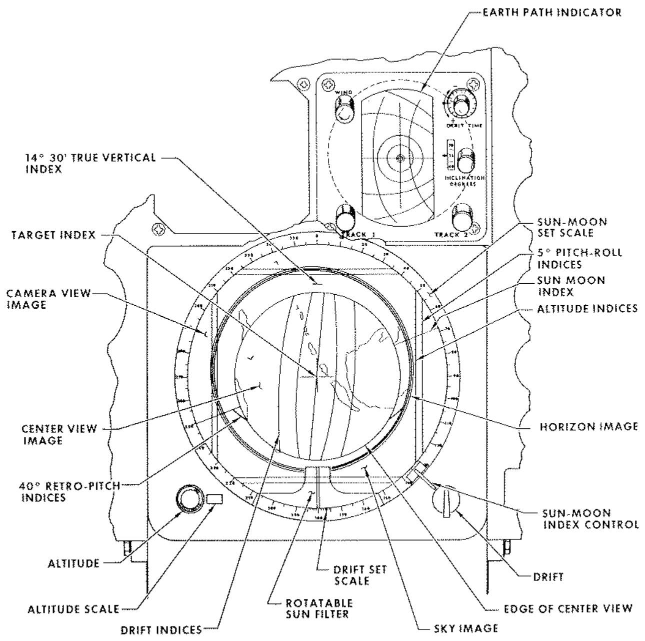

Figure 1: A NASA diagram for the Mercury 5 showing floating labels connected to their anchors as well as break lines: freehand lines are drawn to indicate that the depicted object extends beyond the section shown.

[5]) allow cross-cutting connections inside a single document. The XLibris system also searches for pages containing contant similar to what the user underlined or circled on a page, and suggests links to the related pages as thumbnails in the margin.

In this article, we take the ideas seen in the above references a logical step further, using some visual ideas seen in Fig. 1. We begin from simple design principles and develop a number of new user interface techniques to create a usable interface. These techniques are mostly made possible by modern graphics accelerators, which allow the use of several visual effects on commodity hardware that would have required expensive graphics workstations merely five years ago.

The new visual techniques include link targets floating around the focus called *buoys*; *break lines*, a way of showing and animating a document fragment as a torn-off piece of the whole; and unique background textures for visualising the identity of the documents.

We have named this system BuoyOING (Buoy-Oriented Interface, Next Generation) because the main focus is on the buoys, and this is our second internal prototype of these ideas.

We present an example application that shows PDF documents with connections and annotations using the new techniques. The application is based on xanalogical structure [14] and, orthogonal to it, on a spatial canvas structure that allows the user to enter annotations and transclusions from the documents on a virtual canvas.

In the following sections, we first discuss the user interface techniques in detail, then consider their implementation on the Gzz platform and present our example application. After this, we conclude.

2. THE BUOYOING USER INTERFACE

The design of our user interface is based on three simple principles:

- the user should always see all link targets ("you should see where you can go")
- the link transition should be fluidly animated ("you should see where you do go")
- the link transition and resulting view should make it obvious to the user how to go back, without an explicit back button ("once you get there, you should see how you can get back"). This implies bidirectional links.

Let's start with one scrollable node ("current document"). To be able to show all the link targets near the anchors, only the *relevant fragments* (the immediate surroundings of the other end of the link) of the target nodes are shown. The images of the link targets "float" near the anchors, which is why we call them *buoys*. When traversing a link by clicking a buoy, the buoy expands to become the main view, and the main view shrinks to a buoy.

Fragments of different documents can look very similar, especially preattentively ("at-a-glance") [19]. In order to let the user perceive the source of the fragment, we texture each node with a unique background texture generated procedurally [3, 15] from the node's id.

In the following subsections, we discuss the main components of the interface, buoy placement and unique backgrounds, and following that, some techniques which are not as essential but support this type of interface by clarifying the graphical appearance: break lines, nadir rotations and fisheye.

2.1 Buoy placement

In most current systems, all graphical objects are placed in either the coordinate system of the virtual paper (e.g., the margins of the web page being scrolled) or an external coordinate system, independent of the page (e.g., in another window). This was originally done for performance: updating a single, rectangular (or rectangular, rectangularly obscured) area of screen is most efficient. For the same reason, smooth scaling is only seldom used.

With modern graphics hardware, there is no need to be limited to rectangular sections and discrete scales because the whole screen can be redrawn at interactive frame rates. Thus, it is possible to place objects in different coordinate systems whose motion depends on the others in complex ways.

We use buoys as link targets floating around the focus. What we call *buoy* is a commonly used tool in technical diagrams: placing a label at the edge of the image and connecting the label to the relevant location (anchor) by a line (see Fig. 1).

For the layout of the buoys we give the following criteria (in order of importance):

- buoys should not be placed directly on the focus (center of screen)
- buoys whose anchors are close to the focus should be large
- the view should animate continuously when the focus moves
- the user should be able to understand and predict the motion of the buoys.
- there should be little or no hysteresis (dependence on prior states)
- buoys should be placed close to their anchors
- time coherence: when traversing a link, i.e. animating a buoy into focus, the former focused node should have a clear relationship.

The apparent conflict between "no hysteresis" and "time coherence" can be resolved by maintaining a local spatial structure, giving each link a specific left-right orientation so that a right-end node and left-end node retain their relative locations w.r.t. to the link, independent of either of them being the main node. This matches the way the brain understands space as globally distorted, segmented, locally Euclidian views (see, e.g. [20]).

More than two opposing directions could be used, but this is not required for the local spatial coherence and it would limit the layout of a large number of buoys. Furthermore, there is usually no meaningful global 2D layout for a network of nodes, so the extra directions would not help much in perceiving the position in the global structure.

Orienting the link direction horizontally rather than vertically is more natural because the visual field is wider than it is tall — consider the usual screen aspect ratios of 4:3 and 16:9.

Based on the above principles we have selected a simple geometry depicted in Fig. 2. At the edges of the screen, the right-link-end buoys are rendered at a small scale, placed at a constant distance to the right of their anchors. To avoid placing buoys that would obscure the focus, we define an ellipse somewhat smaller than the screen, and if the buoy would be inside this ellipse, its position and size is calculated differently.

The position is calculated by *projecting* the anchor to the circle from the leftmost point of the ellipse. This places the buoys in a predictable and comprehensible way: the human eye is good at understanding pencils of lines, due to perspective. The size of the projected buoys falls linearly

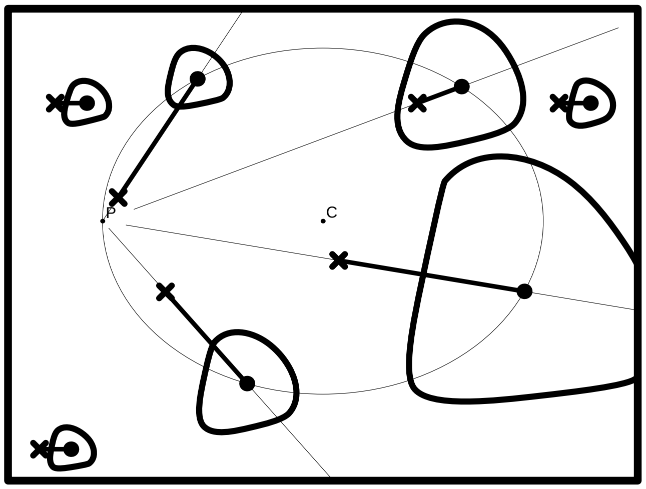

Figure 2: Buoy layout strategy depicted for right-link-end buoys. The crosses represent anchors and the blobs are buoys. Peripheral buoys are drawn at a small scale next to their anchors, while in-focus buoys are projected to an ellipse, retaining the left-right orientation, and shown in larger scale. For left-link-end buoys the layout is mirrored.

with the distance of the anchor from the center up to the edge of the ellipse.

For left-link-end buoys, the mechanism is inversed.

The resulting layout places the buoys close to the anchor while maintaining the left-right orientation. If buoy anchors coincide or are very close to each other the final positions of the buoys need be adjusted to avoid overlap.

2.2 Unique background textures

The fragments of nodes shown in buoys are generally very similar in appearance. The user could identify the nodes by reading the text of the fragment, but that requires too much attention.

Using a unique background texture for each node ("virtually printing the node on fancy paper") changes the situation dramatically: the user can perceive the identity of the most familiar documents at a glance, even when only small fragments are shown. Furthermore, when moving from node to node, the pre-attentive cues of identity help the user maintain a sense of direction.

As the textures are intended for distinguishing *similar* nodes, their appearance should not be made to reflect any features of the content. Instead, the texture should only depend on a permanent id of the node. That way, the textures in any view are similar only by chance and can be recognized (in the sense of at least feeling familiar) even after long periods of time.

The background textures are generated at run time, using the node identity as a seed value to a pseudo-random number generator. Because the algorithm is fixed, a node's texture will remain the same between invocations even though the texture is not stored anywhere.

The generation of usefully unique background textures is not simple - the distribution has to be carefully adjusted to produce maximally diverse and recognizable textures, taking into account the properties of human visual perception. For example, backgrounds with random pixels (noise) would all look the same, because the pixels are not perceived individually. Instead, shapes and overall colors should be used, randomized independently to maximize diversity. Making the backgrounds repeating creates well-defined patterns and improves recognizability when fragments are shown.

Our hardware-accelerated implementation uses a small set of *basis textures*, which are non-linearly combined on the GPU to create a large set of recognizable shapes. The coordinates of the component textures are randomly chosen affine functions of the paper location, but repeating with a randomly chosen *repeating unit* (a parallelogram).

At each pixel, the combined values of the basis textures are used for interpolating between the colors of a small palette of *compatible* colors, randomly chosen from a carefully weighted distribution. That way, the colors and shapes are independently random and the palette can be restricted to light colors to maintain readability.

2.3 Break lines

The rectangular frames used in most user interfaces likely are also a decision resulting mostly from performance. Especially when showing part of a document in a buoy, a rectangular frame could be visually confusing, as it doesn't provide a clear indication whether we see only a fragment of the target or all of it.

Break lines are a technique used in technical drawing for indicating where an object extends beyond what is drawn in the current diagram. It is visually clear since it uses a shape that is obviously not a part of the object's own shape (wiggly freehand line, see Fig. 1).

We apply this technique by drawing the buoys as non-photorealistical pieces torn off the target document. To allow for fluid animation, the shapes of the break lines need to be carefully designed.

The shape of a torn edge is tied to its location on the target document, creating a cue of scale of the torn-off piece on the screen (the wiggly line is scaled with the document when zooming). When a link is followed, the torn shape of the target buoy animates to the full shape of the document. The animation does not look like the edge just gliding over the document, but rather as if larger and larger parts were magically torn off the original document.

The hardware-accelerated implementation uses a noise texture for creating the variation in the torn shape with texture coordinates tied to the paper location. The stencil buffer is used for efficiently drawing the contents delimited by the irregular shape. Finally, a non-photorealistic black edge [17] is drawn around the silhuette to clarify the image.

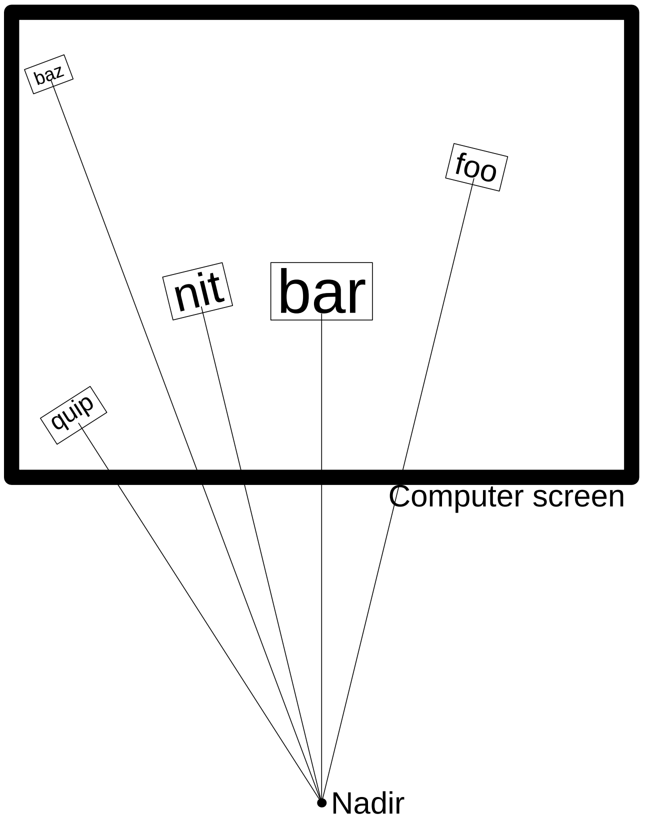

Figure 3: The nadir rotations explained geometrically. The items on screen are rotated so that their local vertical axis points to the vanishing point (nadir) placed outside the screen.

2.4 Nadir

When a person places papers on a desk, they are usually not placed in the same orientation with each other (unless the person is explicitly building a grid). Rather, the papers are arranged so that the bottom edge of each page points towards the viewer, i.e. so that there's a kind of "vanishing point" at the viewer's stomach.

We have found that simulating this on a computer screen produces a visually pleasing appearance: all buoys are rotated so that their virtual y-axis points toward a *nadir* at approximately one screen height below the physical screen.

Interestingly, in our experience, it seems that the placement of the nadir should be at an absolute *angle* below the center of the screen from the viewer's point of view, rather than related to the size of the screen.

Another benefit of the nadir view is that it makes the different pieces of text on the screen visually distinct from each other [19], as demonstrated in Fig. 4.

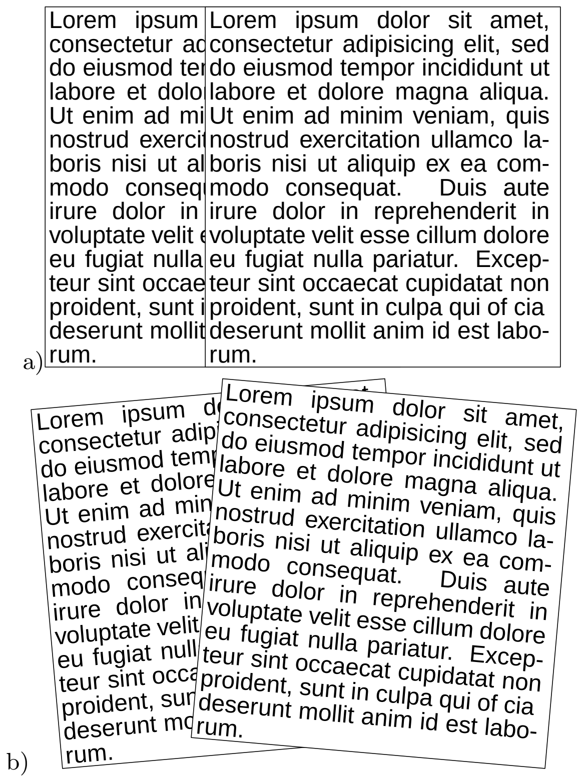

Figure 4: a) two viewports overlapping sideways, with same-size text inside. b) like a), but with the viewports rotated to nadir. In b) the two papers are visually distinguished preattentively

3. IMPLEMENTATION ON THE GZZ PLATFORM

(Note to referees: by the time of the final paper, we will have changed the name of our prototype/platform.) The Gzz platform supports easy prototyping of the above techniques in several ways.

Gzz's Vob graphics model provides a simple way of specifying geometry and automatic animation between views. A vob is a visual object that knows how to draw itself in one or more coordinate systems (for example, a connection line vob draws a line between the origins of two coordinate systems). Views place vobs and coordinate systems into *vob scenes* (rendered keyframes). When the user moves from one scene to another, the coordinate systems of the first view are interpolated to the corresponding coordinate systems of the following view, resulting in smooth animation.

Many source code changes yield immediate effects without rebuilding. Jython source files can be dynamically reloaded and most vobs are specified using strings that are dynamically compiled into OpenGL display lists. Despite this, the framerate is high, because the interpolation of the coordinate systems and the actual rendering of vobs is performed by native C++ code.

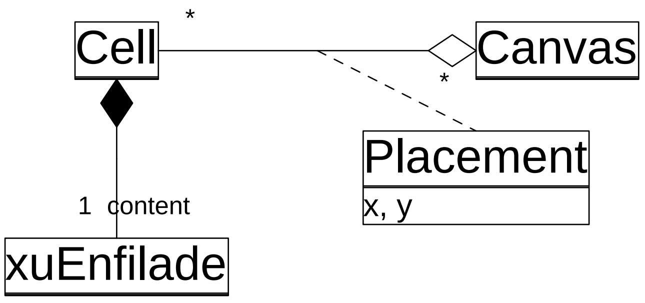

Figure 5: A UML diagram of the two orthogonal structures in the example application. A cell can be connected to another cell by being placed on the same canvas, or by containing an enfilade which is xanalogically connected to the other cell's enfilade.

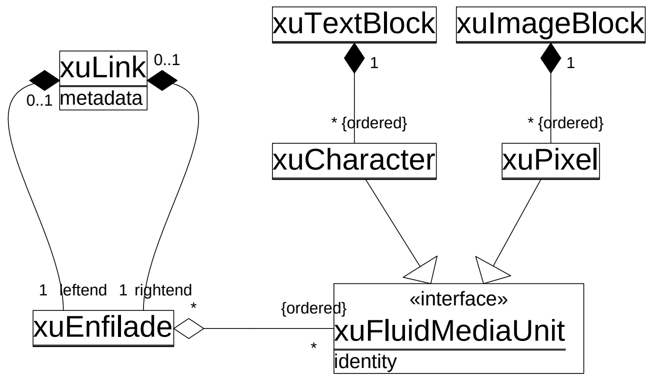

Figure 6: A UML diagram of the xanalogical hypertext structure. The xanalogical connections come about in two ways: if two enfilades contain the same xuFluidMediaUnit, then the two enfilades are connected by a transclusion, and if an enfilade transcludes from the enfilade at one end of a xanalogical link, then it is xanalogically linked to all enfilades transcluding the other end.

The goal of the Gzz project is an environment in which documents from different applications ('applitudes') are seamlessly interconnected. The BuoyOING user interface is a key component of this system. To take advantage of the buoy system, a view only needs to implement an interface for placing itself as a buoy or as the main document, identifying links to show. The system will automatically place the buoys and react to mouse clicks by changing the focus. In the full Gzz system, it will be possible to edit a document with links visible as buoys, follow a link to another document from another application by clicking on a buoy, and continue editing in the second document.

The Gzz platform is free software and can be downloaded through the http://gzz.info website.

4. EXAMPLE APPLICATION

In this section, we present an example application of the above techniques in a user interface for browsing a combined spatial and xanalogical hypertext structure. The intent of the application is to allow the user to create a structure for browsing, annotating and connecting PDF files (for example, academic articles) obtained from other sources.

Our application combines two hypermedia structures: Xanalogical links between pieces of a PDF file [14], and placement of PDF fragments and textual annotations on a 2-dimensional canvas [16, 1]. Overlapping pieces of PDF files can be placed on more than one canvas, creating a visible *transclusion* [13]. A user can collect pieces of PDFs on canvases like in a scrapbook, but with visible connections to the original article.

Internally, the structure is build of *cells* as in Nelson's zzstructure [11]. A cell can have two types of relationships, as shown in Fig. 5: it may be placed on spatial canvases, and it contains a Xanalogical *enfilade* (a collection of Xanalogical media, i.e. a textual annotation or part of a PDF file). The enfilade implicitly connects the cell to all other cells sharing a *fluid media unit* (character or pixel) with this cell (Fig. 6).

Xanalogical structure changes the meaning of 'cut&paste': In the dominant computer paradigm, cutting and pasting moves the actual text characters or image pixels, whereas in xanalogical hypertext, cutting and pasting *copies the references* to the permanent media units (`xuFluidMediaUnit`). Because of this, there is automatically a connection (transclusion) between the original and the copy, and our user interface exploits this by showing the xanalogically connected cells as buoys.

The xanalogical structure also allows for explicit xu Links, which are simply associations between two *enfilades* (lists of references to fluid media units).

The PDF files obtained from external sources fit in the xanalogical structure as fluid media image blocks, and the content of the annotations made by the user are stored in fluid media text blocks. In addition to showing the canvases and the cells on them, the primary application, browsing PDF files, suggests allowing the user to also browse whole PDF image blocks. When browsing a PDF image block, the relevant fragments of xanalogically linked documents and canvases containing a transclusion of the current document are shown as buoys. The PDF block itself is shown using a distortion-oriented Focus+Context view[fc-fisheye-andalso-fc-taxonomy-andalso-carpendale96multiscale-andalso-carpendale01presspace]; the magnification and size of the focus is adjustable by the mouse.

An important point here is that the user interface only shows the structure that is relevant from the user's point of view. A canvas node shows the original PDFs transcluded in the canvas as buoys, but the user is not interested in seeing the constituent media blocks of the annotations. For the same reason the annotations on a canvas do not have unique background textures (as the PDF transclusions do), but the whole canvas has a unique background texture based on its identity.

The left-right orientation of the xanalogically linked buoys is determined by the direction of the xanalogical link. In this prototype, the orientation of a transclusion is fixed so that a transcluding canvas is always left of the PDF image block.

5. CONCLUSIONS AND FUTURE WORK

We have presented several user interface tecniques (some inspired by technical drawing) that together create the non-disruptively linking BuoyOING hypertext user interface. The techniques are made possible by modern graphics accelerators with texture mapping capabilities.

Our prototype demonstrates the techniques on a simple structure consisting of 2D canvases and xanalogical data. The canvas can easily be extended to support "digital ink" annotations.

There are several unresolved issues with this type of user interface. For instance, allowing the user to have several separate "focused nodes" at the same time is necessary to allow linking; there are several possibilities but the choice is important. Also, the buoy algorithms need more work: if there are two buoys simultaneously showing regions of a document close to each other, they should probably be combined. Also, if anchors are very close or coincide, the buoys should be pushed away from each other.

6. ACKNOWLEDGMENTS

The authors would like to thank Tuukka Hastrup, Hermanni Hyytiälä and Vesa Kaihlavirta for discussions regarding this manuscript.

7. REFERENCES

[1] 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(1):3–32, 1996.

[2] B. B. Bederson, J. D. Hollan, J. Stewart, D. Rogers, A. Druin, and D. Vick. A zooming web browser. In *SPIE Multimedia Computing and Networking 1996*, volume 2667, pages 260–271, 1996.

[3] R. L. Cook. Shade trees. In *Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, pages 223–231, 1984.

[4] D. Edwards and L. Hardman. Lost in hyperspace: Cognitive mapping and navigation in a hypertext environment. In R. McAleese and C. Green, editors, *Hypertext: Theory into Practice*, pages 105–125. Oxford: Intellect Limited, 1989.

[5] iMarkup Solutions. iMarkup client. `http://www.imarkup.com/products/imarkup_client.asp`.

[6] A. C. Kay. The early history of smalltalk. In *The second ACM SIGPLAN conference on History of programming languages*, pages 69–95. ACM Press, 1993.

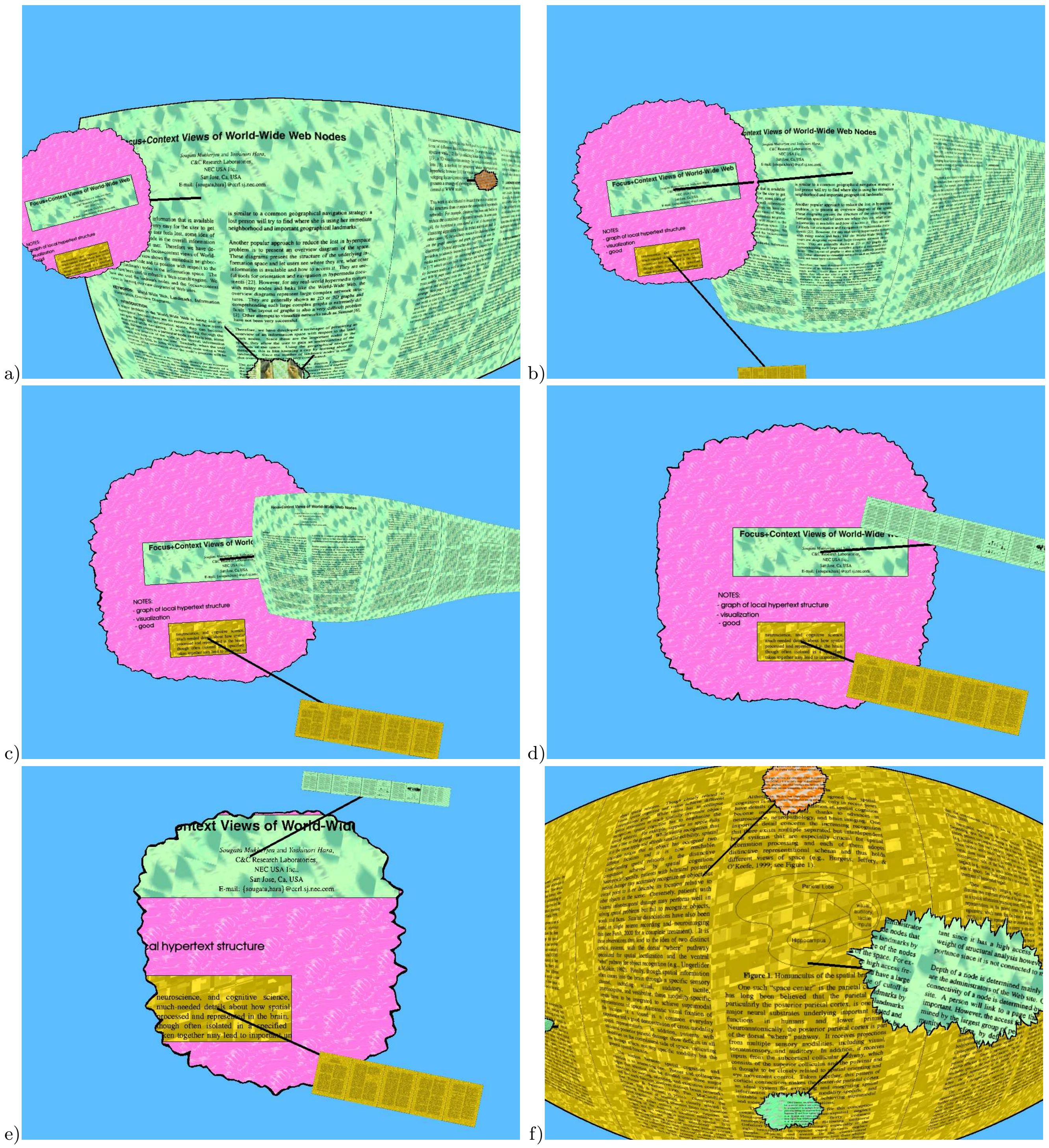
a)
b)
c)
d)
e)
f)
Focus+Context Views of World-Wide Web Nodes
Sougata Mukherjea and Yoshinori Hara,
C&C Research Laboratories,
NEC USA Inc.,
San Jose, Ca, USA
E-mail: {sougata,hara}@ccrl.sj.nec.com
NOTES:
- graph of local hypertext structure
- visualization
- good
neuroscience, and cognitive science, much-needed details about how spatial processed and represented in the brain. though often isolated in a specified taken together may lead to important un
Figure 1. Homunculus of the spatial brain
Parietal Lobe
Hippocampus

Figure 7: Screenshots of the BuoyOING user interface to the example structure described in the text. a) A fisheye view of a PDF document as the focused node and a canvas transcluding it shown as buoy. a) - d) A fluid animation sequence of the transition from the PDF node to the canvas node. d) A canvas view where the user can enter notes and transclude fragments of PDF documents. e) The same canvas zoomed and panned. f) Xanalogically linked fragments of other PDF documents shown as buoys on the PDF node.

[7] P. D. Lebling, M. S. Blank, and T. A. Anderson. Hypertext: An introduction and survey. *IEEE Computer*, 20(9):17–41, 1987.

[8] D. Milgram and W. B. Cowan. Hypercept: behavioural linkage in hypertext environments. In *Proceedings of the 1999 workshop on new paradigms in information visualization and manipulation in conjunction with the eighth ACM internation conference on Information and knowledge management*, pages 49–55. ACM Press, 1999.

[9] S. Mukherjea and Y. Hara. Focus+context views of world-wide web nodes. In *Proceedings of the eighth ACM conference on Hypertext*, pages 187–196. ACM Press, 1997.

[10] T. Munzner and P. Burchard. Visualizing the structure of the World Wide Web in 3D hyperbolic space. In *Proceedings of VRML '95*. ACM Press, 1995.

[11] T. Nelson. Welcome to ZigZag®. `http://www.xanadu.net/zigzag/tutorial/-ZZwelcome.html`, 2000.

[12] T. H. Nelson. As we will think. In *Proceedings of Online 72 Conference, Brunel University, Uxbridge, England*, 1972.

[13] T. H. Nelson. *Literary Machines 93.1*. Mindful Press, Sausalito, CA, 1993.

[14] T. H. Nelson. Xanalogical structure, needed now more than ever: parallel documents, deep links to content, deep versioning, and deep re-use. *ACM Computing Surveys*, 31(4es), 1999.

[15] K. Perlin. An image synthesizer. *Computer Graphics (SIGGRAPH '85 Proceedings)*, 19(3):287–296, 1985.

[16] K. Perlin and D. Fox. Pad: an alternative approach to the computer interface. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 57–64. ACM Press, 1993.

[17] T. Saito and T. Takahashi. Comprehensible rendering of 3-d shapes. In *Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, pages 197–206. ACM Press, 1990.

[18] B. N. Schilit, M. N. Price, and G. Golovchinsky. Digital library information appliances. In *ACM DL*, pages 217–226, 1998.

[19] A. Treisman and G. Gelade. A feature-integration theory of attention. *Cognitive Psychology*, 12:97–136, 1980.

[20] H. Wang, T. Johnson, and J. Zhang. The mind's views of space. In *Proceedings of the 3rd International Conference of Cognitive Science*, pages 191–198, 2001.

[21] P. T. Zellweger, B.-W. Chang, and J. D. Mackinlay. Fluid links for informed and incremental link transitions. In *Proceedings of the ninth ACM conference on Hypertext and hypermedia: links, objects, time and space — structure in hypermedia systems*, pages 50–57, 1998.