A Survey of Zoomable User Interfaces

Winson Chung
Department of Computer Science
University of Toronto
winson.chung@utoronto.ca

ABSTRACT

We present a survey of the evolution of zooming user interfaces (ZUIs) over the last thirty years and issues which exist in preventing their adoption on a wide scale. Using previous work in ZUIs, we will try to illustrate the benefits they provide, and possible future research directions.

In particular, we will highlight the elements of modern ZUIs, and compare the adoption of these elements to full-fledged interfaces such as the PAD interface. By studying these papers, we will be able to see flaws in the design and implementations of ZUIs which greatly affects their overall effectiveness. In addition, we will explore how these issues may possibly be resolved with future research into the topic.

KEYWORDS

Zooming, zoomable, multiscale interfaces, fisheye views, PAD, Jazz.

INTRODUCTION

The amount of information we interact with is becoming increasing complex as more and more technology continues to be integrated into our everyday lives. However, the most basic interface paradigm – the WIMP (Windows, Icons, Menus, Pointer) interface – has not changed significantly from the time it was introduced in the 1980s to account for this need. As such, designers and developers are finding it increasingly difficult to make information accessible through the constraints of both the physical hardware (small screens with low dpi) and existing interaction techniques.

To address this issue, researchers have set out to create an alternative interaction paradigm, known as *zooming interfaces*. This term is used to refer to any physics based interface (or control) which is based on physical interaction (such as panning, zooming, and other geometric transforms) instead of metaphors. This allows for virtually limitless information to be stored, accessed, and retrieved using natural physical actions.

While the term *zooming user interface* (ZUI) was not introduced until Bederson et al [6] in 1994, and are still relatively unknown, zooming interfaces have since matured into having a set of unique identifiable properties. Unlike existing interfaces which are based on common metaphors (such as the desktop), the general zooming graphical interface is based on spatial and physics based interaction in the form of zooming, panning and common geometric transforms [2] which can be applied to an infinite two dimensional information plane.

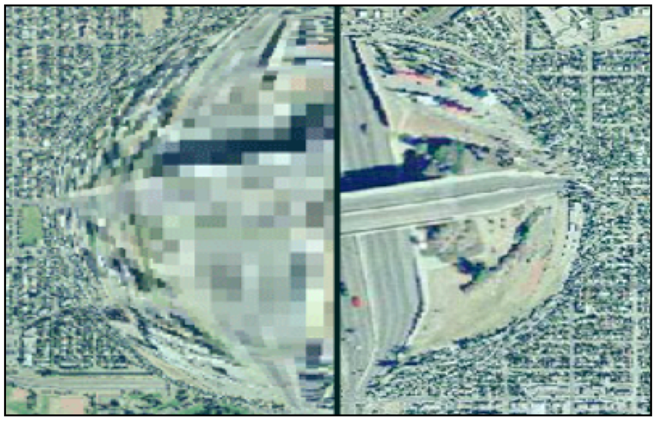


Figure 1. The difference between geometric (left) and semantic (right) zooming.

Generally, each object in the system occupies a specific location, allowing users to make use of their spatial memory in interacting with the interface. While the interface is considered to be three dimensional in nature (magnification along the z-axis), zooming is constrained to the z-axis only, meaning that there is no additional perspective issues – at any point in the interface, the user lies on a normal vector to the information plane. In practice, zooming is sometimes implemented with granularity due to limitations in the data, or other system performance reasons. As we will discover, this is troublesome and often leads to loss of user context.

The Semantics of Zooming

We often refer to geometric zooming in real life (such as a magnifying glass), however, in most featured zooming interfaces, we are actually referring to semantic zooming (see Figure 1). Geometric zooming simply enlarges the view without consideration of the limits of the data (there may be bounds beyond which the information may not display correctly, or legibly). Semantic zooming introduces the notion of multiple representations of information, the displayed representation depending on the state of the view. The use of semantic zooming allows zooming interfaces to provide usable information at the cost of less user control over their interaction process.

Use of Spatial Cognition

A zooming interface allows a user to make use of their spatial cognitive abilities because the system can be represented as a hierarchy in two dimensional space. This allows users to remember relative locations of objects as opposed to absolute paths, similar to the recognition of items on a desktop. This is a key component of zooming interfaces.

THE EVOLUTION OF ZUIs

The idea of modern zooming interfaces owes itself to several key ideas which were developed in the 1980s and 90s.

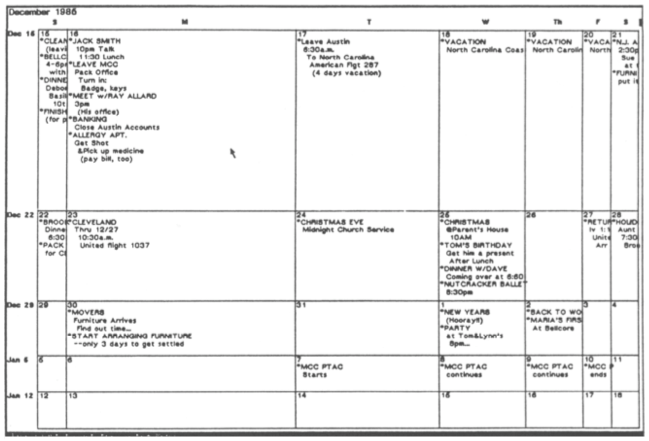
Fisheye Views

The notion of *zooming* first began with early interest in selective representation of computer data through non-linear views from George Furnas [1]. In his paper, Furnas suggests a new method for displaying data of both local and global context using *fisheye views*. His hypothesis was that humans naturally relied on both levels of data to solve trivial common tasks, such as a New Yorker finding the best route to New Jersey (which involves a detailed map finding the best way out of the city, and a more general map to find the best interstate). Because humans are able to translate between these two contexts seamlessly in real life, they were able to perform more complex tasks easier.

He first conducted a number of empirical experiments in which subjects were asked about various common details such as the States and Presidents of America. He hypothesized in his *empirical fisheye conjecture* that the results would contain either items of "great *a priori* importance or be 'close to home'", which was indeed confirmed through the subject's replies.

In formalizing this idea into the fisheye concept, Furnas chose to use a combination of a Distance (D(x,y)) and "A Priori Importance" (API) function, which is the importance of a point to a user. Using these two functions, Furnas' determined the "Degree of Interest" (DOI) from the importance and distance of neighboring items, "assigning each point a number telling how interested the user is in seeing that point, given the task". By displaying more detail for neighboring information, the fisheye view is able to provide the best of both local and global contexts.

Because the distance and *a priori* importance properties could be applied to all sorts of data (not limited to those hierarchical in structure, see Figure 2), Furnas also



December 1986

	S	M	T	W	Th	F	S
Dec 15	15 *CLEAN (leavi *BELLC 4-6p with *DINNE Deboi Basil 10t *FINISH (for p	16 *JACK SMITH 10pm Talk 11:30 Lunch *LEAVE MCC Pack Office Turn in: Badge, keys *MEET w/RAY ALLARD 3pm (His office) *BANKING Close Austin Accounts *ALLERGY APT. Get Shot &Pick up medicine (pay bill, too)	17 *Leave Austin 6:30a.m. To North Carolina American Figt 287 (4 days vacation)	18 *VACATION North Carolina Coas	19 *VACATION North Carolin	20 *VACA North	21 *N.J. A 2:30 Sue at *FURNI put i
Dec 22	22 *BROOK Dinne 6:30 *PACK for Cl	23 *CLEVELAND Thru 12/27 10:30a.m. United flight 1037	24 *CHRISTMAS EVE Midnight Church Service	25 *CHRISTMAS @Parent's House 10AM *TOM'S BIRTHDAY Get him a present After Lunch *DINNER W/DAVE Coming over at 6:60 *NUTCRACKER BALLET 8:30pm	26	27 *RETUR lv 1:1 Unite Arr	28 *HOUD Aunt 7:30 Bro
Dec 29	29	30 *MOVERS Furniture Arrives Find out time... *START ARRANGING FURNITURE --only 3 days to get settled	31	1 *NEW YEARS (Hooray!!) *PARTY at Tom&Lynn's 8pm...	2 *BACK TO WO *MARIA'S FIRS At Bellcore	3	4
Jan 5	5	6	7 *MCC PTAC Starts	8 *MCC PTAC continues	9 *MCC PTAC continues	10 *MCC ends	11
Jan 12	12	13	14	15	16	17	18

Figure 2. A calendar which makes use of the fisheye view to highlight upcoming events.

highlighted example applications of the fisheye view in displaying information such as source code and even English text.

It is important to note that Furnas' fisheye view only supported geometric zooming – since the data remained constant regardless of view size. He also found in his experiments that there were certain cases in which fisheyes had multiple foci which enlarged the local view too greatly. This occurred when there were two similar degrees of interest in multiple locations, and suggested the use of multiple separate views to address this issue.

In terms of zooming interfaces and information visualization, Furnas' paper would provide the foundation for future interest and research into the topic.

Pad

The first truly revolutionary zooming interface was devised seven year later by Ken Perlin and David Fox, under the name of Pad. The interface introduced the concept of an "infinite two dimensional information plane" in which every Pad Object occupied a certain space in the system. By using a spatial metaphor – that of an infinite wall in which a user could read and write at any level (even microscopic), Perlin suggested that the interaction of working with objects which occupy real space is more natural than existing forms. The system relied on heavy use of a user's spatial cognitive abilities to assist in navigating through the interface.

The most interesting elements of the Pad interface were the Portals, Portal Filters, and Semantic Zooming.

Portals represent a view of the Pad Surface (the two dimensional information plane), and each portal can hold different physical properties than the main user view. Perlin gives an example of a financial report in which the main view is zoomed out to fit the entire report, but

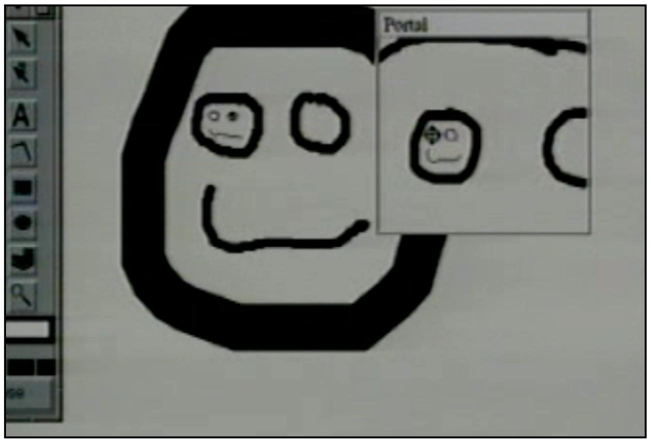


Figure 3. The Pad interface with a portal showing a zoomed out view of the drawing.

several portals remain in order to view certain figures in detail. In the paper, Perlin also stresses the differentiation between a portal and a regular window or terminal, in that the portal, like most of the Pad constructs, is non-dedicated and can be moved to any portion of the screen at any level of detail, unlike windows which are tied to a specific resource.

More interesting are the notion of Portal Filters, which are really an extension of Magic Lens filters conceived by Bier et al [3][4]. When placed over certain objects, Portal Filters modifies the view of the object, displaying an alternate representation. An example would be a bar chart Portal Filter which modifies the view of any tabular data under the lens into a chart. These filters update in real-time, allowing for collaborative modification of underlying data directly, as well as through the filter itself. Like the Portal, the filter can be moved freely from object to object.

Lastly, but perhaps most importantly, the Pad interface was the first to introduce the concept of semantic zooming to the desktop. This allowed certain Pad Objects to display various levels of detail depending on the zoom level of the view under which it is displayed. For example, when zoomed out far enough, only the title of a text document may be shown instead of a scaled version of all the text.

Compared to traditional WIMP interfaces, the Pad interface provided a unique departure in terms of information access and visualization. However, much of the paper focused on the implementation of the interface, as well as possible applications for the technology (such as text editors, painting programs and story books) without any real evaluation the effectiveness of the Pad interface compared to WIMP. However, as an exploratory paper, this research marked the beginning of a long line of full-fledged zooming user interfaces..

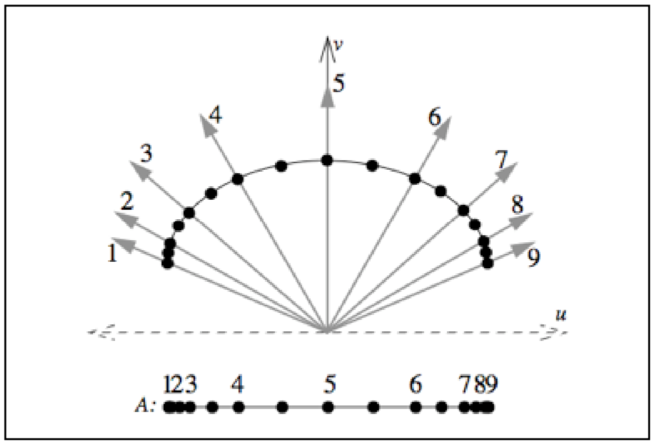


Figure 4. A Space-Scale diagram illustrating the magnification of a fisheye view (one dimensional).

Space Scale Diagrams

Another breakthrough that occurred in the mid 90s was the formation of Space-Scale Diagrams by George Furnas and Benjamin Bederson [5]. In the ZUI research papers prior to the use of these diagrams, the effects of magnification and multiscaling were shown visually through examples and images of the system in use. With Space-Scale Diagrams, researchers were finally able to formally represent both spatial translation and magnification in an easy, straightforward manner. This allowed researchers to analyze the interaction effects of different actions within a zooming interface in both one, and two, dimensions (two dimensions are a simple extension of the one dimensional diagrams, and will not be covered in the scope of this survey).

The beauty of the space-scale diagrams lies in their simplicity. A one-dimensional diagram (like the one in Figure 4) can be represented by the spatial vector *u*, and the magnification vector *v*. The greater the magnification, the farther the point is from the origin. Because the vectors space is infinite, the farther you zoom, the sparser the points (grey vectors denoting there they are at each level of zoom). Likewise, when zoomed out, all the points will be represented as a single point in the view.

In addition to geometric zoom representation, Space-Scale diagrams are also capable of representing semantic zooming by representing different levels of the zoomed object at different magnifications.

Within the paper, Furnas and Bederson were able to apply the Space-Scale diagrams to solve problems of non-linear panning and zooming, as well as to predict optimal shortest paths in scale-space. Informal studies were done to compare the optimal trajectories with traditional panning paths, in which most users preferred the optimal path. Using the methods developed in this paper, Bederson et al [6] were later able to improve their modifications of the Pad interface in developing Pad++.

Pad++

The motivation for Bederson et al [6] to create Pad++ was to incorporate all the research in zooming interfaces into an up-to-date toolkit from which researchers and developers could continue to build interfaces with zooming as a primary interaction.

In particular, Pad++ introduced the notion of *critical zones*, which were adopted from Furnas' previous work with Degrees of Interest in his fisheye views [1]. Critical zones allowed the Pad++ system to determine and bound interesting points of focus, which would then become the zooming points. Selecting within this bound will zoom into the bounded objects, and selecting outside of this bound, you would be taken back to a lower magnification. In addition, the Pad++ system stressed the importance of smooth animation in transitions between pre-zoomed and zoomed states. They found that this animation assisted in keeping users oriented in terms of their context in the system.

By attempting to addressing the issue of loss of context in a zooming interface, Bederson was able to improve the usability of the Pad interface, and create a framework for future improvements. However, as with the original paper [2], there were no empirical studies done to support or disprove the effectiveness of these changes – most of the paper is dedicated to details in the implementation of the toolkit itself.

Jazz/Piccolo

Jazz was created by Bederson and Meyer [8], and is the most recent evolution of the ZUI interface which builds upon Pad++ and other research such as Nested User Interface Components [7]. The paper documents the development of the toolkit (Jazz) using a polylithic design, which makes use of a scene graph (more abstract entities) to compose the scene, rather than the more traditional monolithic design (Piccolo), which uses concrete hierarchies of UI elements and objects (like Swing).

The paper is focused more on the proof of concept implementation, and does not provide any significant research into improving the notion of zooming interfaces. However, it does list a set of common properties which Bederson feels all implementations of ZUIs should support:

- non-rectangular graphics and widgets

Figure 5. Fitt's Law test using Zoom-and-Pick

- ability to rendering complex scenes without significant degradation in performance
- support of arbitrary transforms and hierarchies between objects
- continuous, and animated panning and zooming
- support for semantic zooming
- support for portals (aka. lenses, views)
- support for 'sticky' objects (aka. elements for a heads-up-display)
- support for customization to allow event handling for individual, and groups of objects

All of which are supported by the Jazz platform.

Fisheye Menus

In addition to the development of full-fledged zooming interfaces, there has also been considerable research into the adaptation of zooming for use in existing widgets. A clear example of this is Bederson's adaptation of Furnas' Fisheye Views to linear menus as *Fisheye Menus* [9].

When faced with a long list of menu items in a linear menu (such as the bookmarks menu in a web browser), it is often time consuming to iterate through all the items until you reach the one you are searching for. By using the fisheye view to give an idea of all the menus available while showing only those in focus, users are able to directly select their menu faster.

In the paper, Bederson compared the effectiveness of Fisheye Menus to standard, scrollbar and hierarchy (sub menu levels) based menus. He addressed the issue of jitter movements (disorienting changes in the fisheye view due to small movements of the mouse) by allowing the user to enter a "Focus Lock Mode" in which the menu can be locked before the item is selected.

The Fisheye Menu experiment was mainly qualitative in nature, in which ten subjects were tested, five of whom were Computer Science students (programmers) and five of whom were not. Surprisingly, he found that overall, most people preferred the traditional hierarchical menus, with the programmers preferring the fisheye menus slightly more. Most had issues learning the Fisheye Menu due to misconceptions about the implementation (labels, focus lock issues), but agreed that continual use could make it potentially more effective than traditional menus. Bederson also ran an expert test on himself, in which he found similar outcomes to the preferences determined in the study.

This paper highlighted the possibility of improvement by applying zooming effects to existing WIMP interface controls. However, the author was not able to provide significant empirical data as to the effectiveness of the system in selecting menus, and only times for his personal experiment. From our previous work, this situation may have been easily predicted using KLM-GOMS models. In addition, the author did not address the issue of searching for an item which you did not know the name of, nor the case in which items were not alphabetically sorted. It is possible that future research in this topic could be done to improve the interface such that it would be easier to learn and use.

Zoom-and-Pick

Another application of zooming is the integration of a temporary zooming state within other interface paradigms. An example of this is Zoom-and-Pick, a study done by Clifton Forlines et al [10] who used zooming to solve the problem of jitter and low resolution when using handheld projectors to make selections.

Using a within-subjects design, Clifton et al. conducted an experiment comparing Zoom-and-Pick performance with regular selection using twelve participants and a total of 4032 trials. In the selection-only test, they found

> *a significant main effect for technique on selection error rate ($F_{1,10}$ = 77.51, p < .001), with a mean of 35.6% for regular pointing and 9.2% for zoom-and-pick. ... target width had a significant effect on the selection error rate ($F_{3,10}$ = 56.73, p < .001) ... selection error rate for zoom-and-pick is fairly consistent across all widths, whereas with regular pointing selection error rate significantly increases as width decreased.*

This validated the effectiveness of zooming in reducing error rate and selection complexity. However, Clifton et al. also found that there was a significant main effect for technique on selection time ($F_{1,10}$ = 32.00, p < .001) with a mean difference of 0.50 seconds between regular and zoomed selection, which he attributed to the "added complexity of [using] zoom-and-pick".

In the Select-and-Release task, resulting effects were similarly significant, with Zoom-and-Pick slower with selection and release time on average (0.37s) and a much

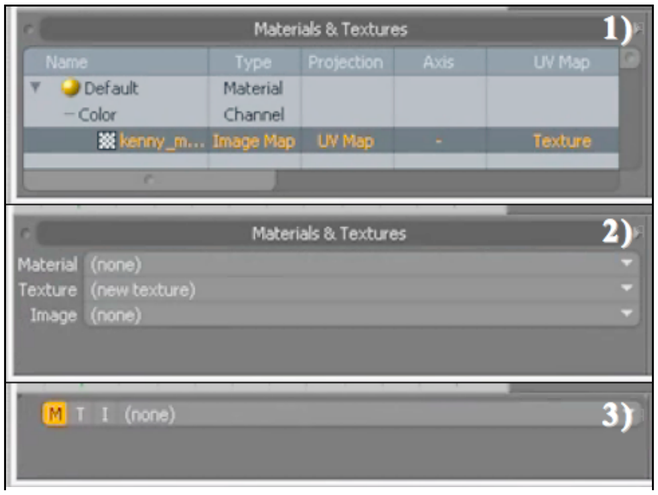


Figure 6. The three levels of detail of the table control in modo™.

improved error rate of only 15.3% for zoom-and-pick, compared with 53.2% for regular pointing.

Overall, this paper was able to highlight the immense improvement in usability which can be achieved by applying zooming interaction to handheld projectors. In particular, the design of the study was very well done, with the author attempting to address various secondary issues such as learning effects, self optimization and subjects tiring. Unlike the previous papers mentioned, this is the first to show, empirically, the effect of zooming interaction compared to interaction within existing interface paradigms.

CLASSIFICATION

In the evolution of the zooming user interface, there have been two approaches to adapting the ideas of ZUIs into mainstream use. The first of which is the full interface approach which has evolved from the Pad [2] interface to Pad++ [6] to Jazz [8] and Piccolo. In terms of the success of these systems, there has yet to be any popular application which is based off these toolkits. However, this is not to say that there is no future for their designs. These interfaces are attempting to solve problems some of which are not yet apparent in traditional UI paradigms. As we begin to reach the limits of existing technology, we may find situations in which traditional UIs are simply incapable of handling – in which case ZUIs may be a viable alternative.

The second approach is more realistic in nature, and already has had an impact on applications in use today. By slowly adapting specific properties of zooming interfaces with existing UIs, we are able to create new forms of interaction which facilitate better information management. Both the Fisheye Menus [9] and Zoom-and-Pick [10] have demonstrated that there is a lot to be gained from this blend of techniques in terms of usability and efficiency. A real world example of semantic zooming in use today can be already be found in the 3D modeler modo™.

In modo™, developers have created a clever technique to maximize view port space (a common request) without completely sacrificing interface functionality. Instead of letting the user control zooming, there are elements of the modo™ UI which automatically react to the resizing of borders as if the user was zooming semantically. For example, in Figure 6, we see three states, the first of which represents the normal UI control which lists an object's materials. However, when the height of the control is resized to be smaller than the lower bounds of the table, it collapses to three dropdown boxes which give the same functionality (though at the cost of additional user input). Finally, if the control is resized such that even the three dropdowns do not fit, then it is reduced to a single dropdown with selection buttons to determine the property that the dropdown refers to. This interaction can be easily visualized using Space-Scale diagrams as a single control which has three states, and resizing the control zooms the control view.

As we begin to hit more and more limitations with existing UI constructs, it is increasingly plausible that we will require newer forms of interaction. By adapting zooming into existing controls, we hope to minimize the learning curve while providing significant gains.

ISSUES

Despite the numerous advantages of zooming interfaces, there are aspects of ZUIs which have yet to be fully addressed in research and implementation. The main issues that plague current ZUIs are:

- the loss of user context
- the difficulty in finding information

and

- the complexity of zooming interaction

Although some issues are not unique to zooming interfaces, they have a greater effect on the usability of ZUIs than traditional interfaces.

Loss of User Context

One of the main problems with ZUIs is that users are able to manipulate the interface in such a ways that they do not know where they are in the global context, and there are no landmarks or clues to guide them back on track. The extreme case of this is called Desert Fog, coined by Juls and Furnas [11] in 1998 in which a user is either zoomed in so far that the whole screen is covered, or they are zoomed out so far that there is no more objects. Within the paper[11], the authors evaluate two possible solutions to the problem: *Landmarking*, and ZTracker.

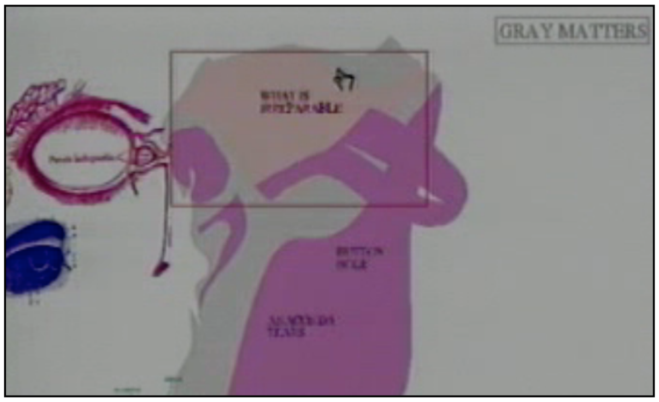


Figure 7. Clicking the Critical Zone will zoom the user in. Clicking the surrounding area will zoom the user out.

Landmarking is the placement of multiple scale-independent landmarks (multiscale residue*)* to represent each object within the system.. A residue is "*a view-navigation term, is evidence that leads a navigator to believe that a particular object may be found in a particular direction*". Because the landmarks are scale-independent, they have the same size regardless of the zoom property of the view. And by adding these landmarks to each node in the hierarchy, users have a simple solution for Desert Fog; if they ever get lost (they can just zoom out until they see a residue. Juls also noted that unless the residue was shown on a different context layer, it is possible for multiple markers to clutter the view of users who are not lost.

A more complex solution, is one which was prototyped in the Pad++ interface, which involve the notion of *critical zones*. A *critical zone* is "*a region of the view where zooming in leads to interesting views*". And by allowing the user to zoom into and out of these critical zones, we can guide users to keep them from making choices which lead to Desert Fog. If they ever encounter a state where there are no critical zones in view, then the computer implicitly knows that the next user action will be to zoom out until one is visible.
Other simple solutions such as limiting a user's zooming capability will also work, but also affects the effectiveness of the interface (the amount of information it can display). While the solutions above provide solutions to the problem, they don't prevent the user from getting into Desert Fog in the first place, which appears to be inherent to multiscale interfaces.

Difficulty in Finding Information

Zooming interfaces also present significant problems in the searching and retrieval of data whose location is unknown – especially if the data is non-textual in nature. This is compounded by the fact that there are often items that one can recognize once they see it, but are unable to describe it in terms of a search query.

A suggested solution to this problem is be the integration of metadata into objects within the system. While this task will grow in difficulty with the number of objects in use, having the additional data will allow the interface to accurately classify and find related items.

Another solution may be some kind of placement strategy (similar to *BumpTop*) in which the distance of related data in the system is close. This ensures that users will have an idea of where to look for specific data, and raises the possibility of innovative uses of portal filters to save and build search queries for reuse as live search results.

A more general problem is the problem of low resolution of consumer monitors in use today (96 dpi is average). This means that the zooming interface is limited, especially since there are objects which may be recognizable on paper, but not on the display.

Luckily, the searching problem is not one that is unique to zooming interfaces, and innovation in that field can be applied directly for use in ZUIs.

Complexity of Interaction

When compared to more common blends of zooming interfaces, pure ZUIs pose significant difficulty in reaching adoption for several reasons.

Firstly, they are difficult to control using existing physical input devices because there are only three buttons on an average mouse, two of which are used heavily on the desktop. For common users to learn to use a zooming interface effectively, they would have to re-learn the buttons on the input device, or replace it out-right.

Secondly, because of the context issue above, zooming interfaces often require animations to occur every time a transition is made. While this is necessary to keep the user in sync with their virtual-spatial location, it affects the response time of the system (user has to watch it every time). This is a disadvantage in the eyes of users who perceive existing interfaces to react instantaneously.

Lastly, a usability study by Hornbaek et al. [12] in 2001 showed that a majority (81.25%) of people still preferred the standard Overview+Detail view (a general view + a minimap) to the purely ZUI view, even though they were more efficient when using the ZUI view in complex tasks. In particular subjects found the Overview+Detail view to be "*easier to navigate*", "*less disorienting*" and "*easier to move [between counties] while at the same zoom level*". Results showed that subjects were up to 22% faster when navigating in a multi-level map when using the zooming interface, with comparable times when navigating and browsing a single-level map.

FUTURE DIRECTIONS

So far, most of the research and development in zooming interfaces has focused on creating interfaces for a single end user. A potential future direction could the use of zooming interfaces in real-time collaborative applications (groupware).

Huahai [13] evaluated this in 1999, and came up with a solution called GroupPad++ which uses Groupkit (a groupware toolkit) to create "a collaborative layer" for Pad++. By incorporating public and private Pads, as well as shared and replicated layers, GroupPad++ provided a straightforward approach to working with non-local partners in a zooming interface.

The author encountered some issues with concurrency and synchronization of objects between clients, which he resolved using a CVS approach of allowing clients to lock system objects based on a token system. Although he found it to be rarely the case where two people would work on the same item, he describe a situation in which changes to the same object would result in modification conflicts, which it would resolve by discarding someone's changes in order to ensure compatibility between multiple clients (also the case for synchronization of locally cached objects). Huahai suggests that further research will be necessary to improve the transparency and effectiveness of a groupware based zooming interface.

In addition to the implementation aspects of the paper, Huahai also addressed the issue of collaborative awareness between distant users. An important aspect of which is the notion of user space, which defines the bounds with a user is interacting with the system. However, there is little mention of the problem of multiple users who need to work on the same objects under different level of detail, and how collaborative ZUIs might deal with the issues.

This highlights some of the intricacies of zooming interfaces and the possibility of greater research into their development for collaborative uses.

EVALUATION METHODOLOGIES

The zooming interface has yet to fully mature, and much of the research and development has been exploratory in nature, rather than experimental. As a result, most of the current papers regarding ZUIs only address their implementation and innovation without empirical evidence to the validate some of their claims. Another reason for this may be the difficulty in finding unbiased tasks which equally test both purely zooming and standard interfaces.

I feel that there is great potential for integrating aspects of zooming interaction into existing UIs, an example of which was the Fisheye Menus [9]. And in these instances, it may be possible to use techniques such as GOMS analysis to predict the possible results, which would be possible due to the linear and simple actions which are being quantified (panning, zooming). These directions of research would also yield greater experimental support for zooming interaction.

CONCLUSION

This paper was an attempt to summarize the state of research and development of zooming user interfaces as an alternative to traditional WIMP interfaces. We have identified a number of key papers which provide the basis of continual research on the topic, as well as inspiration for new forms of interaction.

In addition to the benefits of purely zooming interfaces, such as increased information and seamless context switching, we also found issues which currently prevent the adoption of the technology on a wide scale. These problems, such as loss of context, difficulty in locating information, and the additional complexity of using a zooming interface, are issues all which require future research in order to resolve.

The notion of zoomable user interfaces has matured greatly over the past thirty years, and I expect to see a more interfaces taking advantage of the zooming interaction as we begin to run into limitations with existing UI paradigms. At which point, more experimental research will be conducted in validating some of the results and claims which have been made so far.

REFERENCES

1. Furnas, G. W. (1986). Generalized Fisheye Views. In *Proceedings of Human Factors in Computing Systems (CHI 86)*, ACM Press, pp. 16-23.
2. Perlin, K. & Fox D. (1993). Pad: An Alternative Approach to the Computer Interface. In *Proceedings of Computer Graphics (SIGGRAPH 93)*, ACM Press, pp. 57-64.
3. Bier, E., Stone, M., Pier, K., Buxton, W., & DeRose, T. (1993). Toolglass and Magic Lenses: The See-Through Interface. In *Proceedings of Computer Graphics (SIGGRAPH 93)*, ACM Press, pp. 73-80.
4. Stone, M. C., Fishkin, K., & Bier, E. A. (1994). The Movable Filter As a User Interface Tool. In *Proceedings of Human Factors in Computing Systems (CHI 94)*, ACM Press, pp. 306-312.
5. Furnas, G. W. & Bederson, B. B. (1995). Space-Scale Diagrams: Understanding Multiscale Interfaces. In *Proceedings of Human Factors in Computing Systems (CHI 95)*, ACM Press, pp. 234-241.
6. Bederson, B. B., Hollan, J. D., Perlin, K., Meyer, J., Bacon, D., & Furnas, G. W. (1994). Pad++: A Zoomable Graphical Sketchpad for Exploring Alternate Interface Physics. In *Journal of Visual Languages and Computing*, 7, pp. 3-31.
7. Perlin, K., Meyer, J. (1999). Nested User Interface Components. In *Proceedings of the Twelfth Annual ACM Symposium on User Interface Software and Technology (UIST 1999)*, pp. 11-18.
8. Bederson, B. B., Meyer, J., & Good, L. (2000). Jazz: An Extensible Zoomable User Interface Graphics Toolkit in Java. In *Proceedings of User Interface and Software Technology (UIST 2000)*, ACM Press, pp. 171-180.
9. Bederson, B. B. (2000). Fisheye Menus. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST 2000)*, pp. 217-225.
10. Forlines, C., Balakrishnan, R., Beardsley, P., Baar, J. v., Raskar, R. Zoom-and-Pick: Facilitating Visual Zooming and Precision Pointing with Interactive Handheld Projectors. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST 2005)*.
11. Jul, S. & Furnas, G. W. (1998). Critical Zones in Desert Fog: Aids to Multiscale Navigation. *(UIST 1998)*, ACM Press, pp. 97-106.
12. Hornbaek, K., Bederson, B. B., & Plaisant, C. (2001). Navigation Patterns and Usability of Overview+Detail and Zoomable User Interfaces for Maps. *ACM Transactions on Computer Human Interaction (TOCHI 01)*, 9 (4).
13. Huahai, Y. (1999). Collaborative Applications of Zoomable User Interface. *Collaboratory for Research on Electronic Work*, School of Information, University of Michigan.

PROMINENT RESEARCHERS

Benjamin B. Bederson

Bederson is currently an Associate Professor of Computer Science and Director of the Human-Computer Interaction Lab at the University of Maryland, College Park. And is one of the most prominent researchers in the field of zoomable user interfaces.

His most significant contributions to the field are Pad++[6], Jazz [8], and Fisheye Menus[9], along with numerous papers which detail the use of his ZUI toolkits in applications such as digital photo management (PhotoMesa, 2001), web browsing, painting (KidPad, 2002) and presentation tools (CounterPoint, 2002).

In addition, Bederson is one of the few researchers who are still actively developing Jazz and Piccolo – the latest incarnation of a full-fledged zooming interface toolkit.

George W. Furnas

Furnas is currently a Professor of Computer Science and Associate Dean for Academic Strategy at the School at Information in the University of Michigan. He is a member of the CHI Academy, and his principle focus is on information access and visualization. His earlier papers have had an incredible impact on the development and research of zooming interfaces today.

His most significant contributions to the field come from his papers on Generalized Fisheye Views [1], and the use of Space-Scale Diagrams [5] to represent spatial and magnified properties of objects. These two papers have been heavily cited in research relating to zooming interfaces and information visualization in general.

Currently, Furnas is working on resolving the challenges of "adding multiscale capabilities to collaborative virtual environments", in additional to work in other computing directions.

Ken Perlin

Perlin is well known in the computer graphics industry for his work on Perlin noise, Hypertexturing, and real-time interactive animation, the first of which earned him an Academy Award for Technical Achievement in 1997. He is currently a Professor of Computer Science and Director of the Media Research Laboratory at the New York University where he studies computer graphics and human-computer interfaces.

In the field of zooming interfaces, his most significant contribution comes from the creation of the Pad interface [2] with David Fox which has become the foundation for modern ZUIs. In addition, his work has iteratively improved the state of zooming interfaces since then, through ideas, such as Nested User Interface Components [7], which would be implemented in modern ZUI toolkits such as Jazz [8] and Piccolo.

Other notable researchers in the field include David Fox, Jon Meyer, and James D. Hollan, all of whom can be found in recurring papers regarding zooming interfaces.