The Promise of Zoomable User Interfaces

Benjamin B. Bederson
Human-Computer Interaction Lab, Computer Science Dept., University of Maryland
3171 A.V. Williams Building, College Park, MD 20742
www.cs.umd.edu/~bederson

ABSTRACT
Zoomable User Interfaces (ZUIs) have received a significant amount of attention in the 17 years since they were introduced. They have enjoyed some success, and elements of ZUIs are widely used in computers today, although the grand vision of a zoomable desktop has not materialized. This paper describes the premise and promise of ZUIs along with their challenges. It describes design guidelines, and offers a cautionary tale about research and innovation.

Author Keywords
Zoomable User Interfaces, Zooming User Interfaces, ZUIs, multiscale, information visualization.

ACM Classification Keywords
H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

INTRODUCTION
The essential problem that Zoomable User Interfaces (ZUIs) aim to solve is a fundamental one – that there is more information than fits on the screen. The common solutions to this problem are, roughly, scrolling, linking & searching, along with denser representations (i.e., information visualization). Zooming, like fisheye displays [21] is an instance of the latter – a kind of information visualization that aims to take advantage of human spatial perception and memory. ZUIs place documents in two-dimensional space at any size, enabling (and requiring) animated spatial navigation to move among documents.

What is a ZUI?
Before we go further, let us think a little about what it means to be a ZUI. Many applications include some kind of visual scaling functionality. Modern file browsers let users control the size of icons.

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

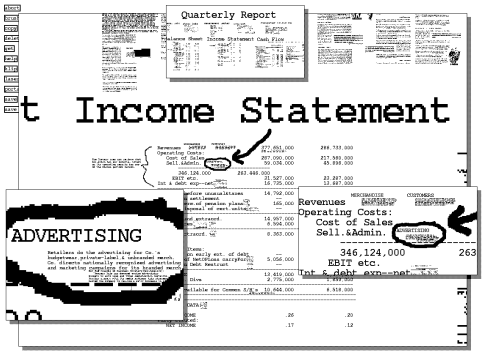

Figure 1: Early ZUI: Pad shows content at different sizes with portals that show a remote region of the data surface (1993) [41, figure 1].

Figure 2: Recent ZUI: Zumobi's ZoomCanvas zooms in from the entire canvas on startup. Dragging left/right pans, and tapping on a region zooms in for interaction with the detailed content (2009) [54].

Web browsers, word processors, image editors, and in fact, almost all full-featured document editors and browsers let the user control the magnification of the document. Many let the user zoom far enough out to see small thumbnails of all the pages of even long documents on modest size screens. However, that kind of simple scaling is outside the scope of this paper.

This paper defines ZUIs to be those *systems that support the spatial organization of and navigation among multiple documents or visual objects* (examples in Figures 1 and 2). Admittedly there is a gray area where it might not always be clear whether a particular application is a ZUI according to this definition. For example, a word processor or document viewer that lets you zoom out and see thumbnails

of pages laid out in a grid minimally meets that definition. However, the pages are really elements of a whole, and not movable in space individually. On the other hand, a viewer that let you zoom out, see and move arbitrary numbers of separate documents, even that were one page each, would count as a full ZUI according to this definition.

According to Cockburn et al.s' survey [18] of approaches to fitting information on the screen, ZUIs display information that is *temporally* separated. The essence of this approach is that the user moves through space and builds up a spatial model of the information in their head. This is distinguished from *spatial* separation (found in overview+detail interfaces such as those found in maps) and *focus+context* or "fisheye" distortion such as that found in the Apple OS X Dock [1] and with the tabular approaches of TableLens [44] and DateLens [13]).

Why ZUIs Excite People

Based on my own analysis of the literature, I have identified three key characteristics that have attracted people's attention over the years. The promise of ZUIs comes largely from the following general expectations.

ZUIs are engaging. The animation is visually attention grabbing. It takes advantage of human visual perception abilities. People can process "visual flow" [52] pre-consciously so it feels easy to build a mental map of the information.

ZUIs are visually rich. There are more degrees of freedom to visually structure objects, and thus they offer the potential of great creative expression. This was identified by Perlin and Fox in their original paper on Pad with their example of a branching tree story [41].

ZUIs offer the lure of simplicity. The fact that you find information by looking for it in a place implies a promise of simplicity that will solve our organizational and information retrieval problems. The overview one sees when zoomed out seems like a panacea – that one finally knows where everything is. This idea was captured in the conclusion of Perlin and Fox's original paper: "As compared to standard current window models, this system makes it easier for the user to exploit visual memory of places to organize informationally large workspaces." [41].

Yet, with these qualities, the grand vision of a zoomable desktop has never been broadly achieved. For example, several commercial efforts have disappeared (e.g., Cincro Zanvas, GeoPhoenix Zoominator, and Innovative iBrowser). Variations such as Task Gallery [47], which used linear zooming in a 3D environment among others also have not been broadly used. There has, however, been a resurgence of commercial interest in ZUIs recently, but as we will see they have significantly scaled back expectations as to where zooming can be useful.

ZUI'S PREMISE AND PROMISE

ZUIs have interested researchers since they were introduced in 1993 by Perlin and Fox [41]. There have been a number widely cited *systems* (e.g., Pad [41], Pad++ [6], Jazz [10] and Piccolo [12]), *applications* (e.g., PadPrints [26], PhotoMesa [11], CounterPoint [23], KidPad [20], AppLens & LaunchTile [32], Seadragon [38], Fly [35], pptPlex [37], Canvas for OneNote [36], and Prezi [43]) *theoretical constructs* (e.g., Space-Scale Diagrams [22] and Desert Fog [30]) and *studies* (e.g., [14, 19, 29, 33]) to support and understand them. Many of these have had a fair amount of accolades and positive user response, yet it is fair to say that none of them have been great commercial successes.

In fact, I would argue that ZUIs have never reached the level of broad use envisioned by their original creators. There has been a huge amount of effort relating to ZUIs with dozens of published papers and a commensurate amount of technology developed. Zooming has been successful in that some kind of zooming is commonly used in a wide range of interfaces (e.g., Google Maps, Microsoft Word, and Adobe Photoshop.) But a richer kind of zooming that takes over the desktop and becomes the primary interface to one's data never materialized.

As one of those early creators, and as someone who has worked on many aspects of ZUIs since nearly their beginning, I think it is worth reflecting on what that original excitement was about, where ZUIs have been successful, where they haven't been and why. For example, a number of commercial efforts that pursued a deeper kind of zooming had only modest success (e.g., an effort by Sony to include zooming in early VAIO computers [34, pp. 160-165], and Hillcrest Labs' HōME product that has not had significant distribution [27].)

In reflecting on this body of work, it may be possible to increase our understanding of why it is sometimes harder to bring innovations to broad use than initially thought. And as researchers, why we should perhaps be a bit more tempered in our enthusiasm – while, of course, not stifling innovation before it has the opportunity to flower.

The Early History

"Pad" was the first system that explored this space [41] (Figure 1). Pad ran on a Sun SPARCstation with black and white graphics, displayed one bit per pixel bitmaps, and used non-animated "jump" zooming (each mouse click would redisplay the view magnified or reduced by a factor of two. Pad offered navigation, authoring, semantic zooming and portals. The term "semantic zooming" (coined by David Fox) refers to how objects can have different visual representations at different sizes. Portals were rectangular objects on the surface that acted like cameras that showed other parts of the surface. Portals were designed to solve the limitation that spatial layout implied. It enabled objects that were physically far apart to be used near each other.

Back in 1992, I was finishing my PhD at NYU, watching Perlin's work closely, and when I joined Bellcore after my graduation, I began building the first of what turned into a series of successors to Pad. While working with Jim Hollan, I started with Pa3D, a ZUI with richer vector and bitmap color graphics and smoothly animated zooming – running on much more expensive SGI computers. Notably Pa3D was in a 3D environment where every polygon could be a zoomable surface. There are no papers about Pa3D, but a video showing it is available online [5].

Pad and Pa3D assumed that information would be placed at different scales in this gigantic information space, and users would access the information by panning and zooming through the space using portals to connect otherwise distance objects. In other words, we aimed to build a zoomable desktop. We made many demonstrations of how people might organize their information in this space, but notably, despite our conviction that zooming was an incredible solution, we had only limited examples of how people would perform day to day tasks.

Being confident that zooming was a good idea, but just not sure what for, I started building Pad++ also with Jim Hollan [6, 7], a successor to Pa3D, that incorporated a well-defined API so others could build zoomable applications, and hopefully reveal the power that zooming offered.

Over the years, teams I led (with Jon Meyer, Jesse Grosjean and Aaron Clamage) extended Pad++ and then built ZUI platforms Jazz [10] and Piccolo [12, 42]. We supported other platforms and languages, and even supported mobile devices with PocketPiccolo.NET. There was also a significant effort to support smooth zooming, high quality animated graphics, and overall high performance on what at the time were fairly meager computing systems [9]. Many other researchers, and a few companies built extensions and applications on these platforms (especially Pad++) as did we. In more recent years, commercially available platforms such as Java, .NET and Flash made it easier to build zoomable applications, and client/server solutions have also been built [38, 53].

One significant effort was by Sony starting in 1998. They started an internal effort to produce ZUIs for their VAIO PC computers which were new at the time [34, pp. 160-165]. Franklin Servan-Schreiber who led that effort coined the term "ZUI"[1], and is the person leading the current effort at Zoomorama [53]. However, Sony never shipped any ZUI products related to that effort.

Looking at the core original ZUI ideas (zooming, semantic zooming and portals), it is interesting to see which ones have had broad usage. Basic zooming has clearly been used very widely in everything from maps to document viewers. Semantic zooming has also had some broader use – even outside the realm of ZUIs. The "ribbon" in some Microsoft Office products (such as Word) use semantic zooming to display more information when more space is available. They dynamically change as you resize the ribbon to show more buttons, and to show more imagery as the panel gets larger. Similarly, several onscreen music players show more controls and information when the player is enlarged. It is notable that not a single one of these applications use portals. Many applications (such as Microsoft Word, Google Maps, and Adobe Photoshop) offer multiple views or a small fixed navigation overview. However, none of these applications that focus on zooming as a core organizational and navigation technique use portals in a way similar to how they were envisioned.

ZUI Characteristics

There are a few essential characteristics of ZUIs that affect their usefulness and their usability.

By *layout flexibility*, I refer to how much creative control the end user has over the layout of information in the space. On one hand, users can put anything at any size and place. This yields complete artistic freedom, but can be difficult to author – and has the potential for creating a visual mess that users can get lost in. On the other hand, the environment might be constrained allowing information only in specific places according to a grid or layout algorithm. A related characteristic is whether specific layouts are *multi-level*, by which I mean that objects in the space appear at significantly different sizes requiring a user to navigate to different zoom levels in order to see all objects.

By *navigation mechanisms*, I refer to how users move through the space. Again there is a tradeoff between flexibility and usability. Some interfaces allow users to fly through the space going absolutely anywhere – including deep into the spaces between objects (resulting in some researchers to label this phenomena *Desert Fog* [30]). Very few other applications let a user navigate beyond the actual content. Almost every document browser and editor limits navigation to the available content (with the notable exception of Microsoft Excel's scrollbar arrows – Apple Numbers and Google Spreadsheet do limit navigation.) On the other hand, some interfaces allow you only to click on objects to zoom into them and click on a zoom out button to zoom out – making it impossible to get lost, but also giving less control over exactly where you look.

One approach to managing the complexity of navigation is to zoom automatically when needed using a technique called Speed Dependent Automatic Zooming (SDAZ).

Speed Dependent Automatic Zooming (SDAZ)

One of the reasons that navigation is such a big issue is that there are too many degrees of freedom to easily control. In a traditional computer interface, one button is typically used

[1] Some people say "zoomable user interfaces", and some say "zooming user interfaces". I prefer the former since it focuses on the fact that they are under user control and don't move around on their own. However, there isn't a substantive difference between these terms.

for selection or action, and a scrollbar or specialized key is used for navigation. ZUI users, however, must pan along two axes instead of just one, and they must be able to zoom in and out. For most applications, this must be done with the same hardware used by non-zoomable interfaces.

Some researchers addressed this by reducing the number of navigational degrees of freedom that users needed to control in the first place [17, 28, 49]. One of the reasons for zooming out is simply to make it faster to pan a large distance (as explained with space-scale diagrams [22] and elaborated on by Van Wijk and Nuij [51]). This observation led to SDAZ where the application would automatically zoom out when the user panned quickly, and then zoom back in when the user slowed down.

Several studies showed the effectiveness of this approach [17, 49], and yet not a single research or commercial ZUI application (outside those that are explicitly exploring SDAZ) uses SDAZ as even an optional navigation mechanism. Why is this? There isn't a clear answer, but it is important to observe that while SDAZ simplifies one thing (degrees of freedom of control), it complicates another (device control and perception).

SDAZ requires the user to engage in a real-time perceive-think-act loop [15]. This real-time interactive manipulation was typical of the "flying" in many of the earliest ZUI systems – and that caused significant usability problems. People that play video games like the interaction challenge of real-time interaction. People doing information processing tasks typically don't. They are more likely to want to spend their cognitive resources on the underlying task, not on analyzing the interface.

Touch Screens and Multi-Touch Interaction

The use of multi-touch "pinch" gestures on touch screens offers an alternative mechanism to control zooming that avoids some of those navigational challenges. Ishii and Ullmer originally envisioned this approach by indirectly using physical objects to scale and rotate images on a screen in 1997 [30]. Then, in 2002, Rekimoto developed a direct solution by using multiple fingers on a capacitive touch screen to scale and rotate images [46]. Finally, this approach was commercialized and is now broadly used in Apple's iPhone and other mobile devices as well as Microsoft Surface. Further, support for this kind of multi-touch interaction is expected in the upcoming Microsoft Windows 7 operating system and may become more common even in laptops and netbooks. This kind of direct multi touch interaction is easy to learn and operate. Thus, while it seems likely that zooming will be used more broadly with multi-touch input, it isn't clear if it will spread to devices without multi-touch input.

Applications

To see something of the range of uses of ZUIs, and how their characteristics have changed over time, Table 1 shows a selection of zoomable applications. Only "true" ZUIs are shown (those supporting more than one document or object.) It captures a range of what people have been using zooming for, and makes apparent the range of approaches that people have taken with regard to layout flexibility and navigation. It is also clear that the essential problem of getting lost in desert fog has not be consistently avoided. Furthermore, it is clear that there is no consistency in the mechanisms that are used to navigate through space.

Studies

Some of these and other ZUI applications have been studied for usability or effectiveness. However, it is the nature of those application-centric studies that it is difficult to tease out the relative benefit or cost of the zooming characteristics of those applications.

In fact, there are relatively few studies that look at the value of animated zooming more generally. But a few are particularly useful, so let us look at them.

Overviews

The first study by Hornbæk et al. looked at multi-scale information structure as well as the use of small overview windows that show a zoomed out view of the information [29]. Surprisingly, it found that while participants like the overview window, its use resulted in lower performance. The study used geographical maps as the data set. Navigation worked by dragging the mouse to pan, holding down the left button to zoom in, and holding down the right button to zoom out. The study varied whether or not users were shown overviews, and whether the map information was single or multi-level. That is, single level maps displayed all textual information in a single font size, so when the user zoomed out, the text could not be read. Multi-level maps displayed textual information at different sizes. Text referring to larger areas on the map was displayed at much larger fonts so that when users zoomed out, they could read the broader-scale text.

Before this study, it was believed that overviews were a good idea and improved subjective satisfaction [40] and efficiency [3]. However, this study found a discrepancy between preference and performance. Study participants preferred the overview condition, but actually performed better in the no-overview condition, especially when using the multi-level map.

It turns out that switching between overview and detail windows was correlated with higher task completion time, suggesting that integration of overview and detail windows require mental and motor effort.

This result is notable because a fundamental concern about ZUIs is that they require time to use (since the animations take time to occur). Worse, there is the potential that ZUIs tax human short-term memory (STM) because users must integrate in their heads the spatial layout of the information. On the other hand, there is the hope that the animated transitions are understand pre-consciously by the human

visual system, and thus may not in fact place a significant load on STM.

This study gives some evidence that the cost of understanding an animated zoomable map of information is in fact less than the use of one with an overview which is one of the primary alternative interface designs. Of course it doesn't say anything about the benefit or cost of a spatially organized information space compared to a non-spatially organized space. It also doesn't say anything about the benefit or cost of animation.

Animation

A second study looked at simple animated navigation transitions and found that these animations significantly increased performance – even including the time the animations took. These kinds of animations have long been expected to provide benefit [16], but evidence backing up this understanding had been lacking. Klein & Bederson [33] looked at a very simple but highly controlled use of animation when scrolling while reading documents. Understanding animation precisely is important because the potential benefit of animation is undermined by the fact that animation by its nature takes time. So, even if the animation provides some benefit, there is a cost as well, and it isn't obvious whether the benefit outweighs the cost.

Participants in this study read documents out loud and when they got to the end of a page, they scrolled by pressing the down arrow key. The animation speed varied between 0 (i.e., unanimated), 100, 300 and 500 ms. The document type was also varied between unformatted text, formatted text, and abstract symbols (where participants counted symbols instead of reading). The reason for varying the document type was to understand how visual landmarks interact with animation.

The essential result was that animation did in fact provide a significant benefit, even considering the time spent on the animation. Reading errors were reduced by 54% with 500 ms animations. Reading task time was reduced by 3% and counting task time was reduced by 24% for 300 ms animations. The formatted documents had a higher overall performance (and lower improvement), implying that visual landmarks appear to be a good idea.

The lesson from this study is that when designed correctly, the benefits of simple animated navigational transitions can outweigh their costs. This supports the design decision to make zooming transitions use short animations. This study is not definitive however since it is possible that the benefit that was seen while scrolling would not occur while zooming. However, there is no obvious reason that the same benefits wouldn't occur, so this kind of animated transition continues to seem valuable.

Presentations

Two other studies looked at how people learned from zoomable spaces [14, 23]. The outcomes here were more mixed, where the zoomable presentation did not produce better learning or memory of the presentation – although participants did remember the structure of the presentation better. The first study looked at how children responded to a story that was presented with or without animated zooming. The children with the animated condition elaborated more during a discussion, but did not otherwise recall the story better. The second looked at how college students responded to a slide presentation again with or without animated zooming. These students also did not recall the presentation better, but the ones that saw the animated zooming condition did recall the *structure* of the presentation better.

Together, these studies imply that zooming alone is probably not enough to help people remember content better. However, users of ZUIs may be more engaged, and they may remember the spatial structure of the content better – even if they don't remember the actual content itself any better.

DISCUSSION

A trend in ZUI applications that appears to be fairly clear over the years is that over time, these applications have tended to get simpler and easier to use while at the same time offering less creative control to end users. This is an important trend because researchers often have a tendency to create tools which are rich and powerful. After all, it is easier to conceptualize an innovation as offering "more" rather than offering "less".

But the market place is clear – success often requires simplicity first and power second. Of course, marrying the two where the power is under the hood is often the best solution – but also very difficult to design. The success of so many modern applications (e.g., search, twitter, photo sharing, iPhone App Store) are examples of this in that they can be used with almost no training in tiny little bursts.

I used to think that only public kiosks had to have these simplistic characteristics – but now it seems that even applications which people use for hours a week and become true experts in follow the same rules.

If true, this implies that we as researchers have to strongly consider not only what new things technologies can help people do – but also how simply and quickly as fundamental goals.

Why ZUIs Are Challenging

As summarized in Table 1, the potential benefits of ZUIs are sometimes mirages. ZUIs are generally engaging (although they make some people feel physically sick) and visually rich. But the promise of simplicity falls short.

While human visual perception does make it easy to see where one is navigating, the reality is that it places a heavy load on short term memory to remember where in space you just were and where things are. And the requirement of human memory to know how space is organized means that

ZUIs don't scale up very well. ZUIs are often motivated by the physical world and how people like laying papers out on their desk. But no one wants *all* of their papers on their desk. It is much more common to have only a relatively small number of papers that are actually being worked with.

The visual overviews that ZUIs offer for free by zooming out may seem like a solution to the load on human memory, but in practice it doesn't because visual overviews of any complexity require significant scanning and visual search in order to find anything. If there are just a small number of objects, then the visual search task is not hard – but of course, for a small number of objects, you don't need a ZUI to solve your organizational problems.

Finally, the visual richness of ZUIs is a double-edged sword. It requires skill to design a complex space with documents of arbitrary size, aspect ratio and color that people can comprehend and scan. Also, people are not as good at scanning 2D designs as 1D layouts – unless the layout is highly structured. But highly structured 2D layouts don't work well for visual objects of arbitrary aspect ratios. Designers are obligated to leave a lot of unused space, scale down the large objects so they are unreadable, or crop the large objects – thus losing much of their distinguishability.

Promise	Reality
Engaging	If designed well.
Visually rich	Can also make space more complex to navigate and understand.
Simplicity	Doesn't scale to very large datasets.

Table 2. Promise and Reality of ZUIs.

Design Guidelines

So how should ZUIs be designed to best overcome these obstacles? Let us start with the big issues.

Support the right tasks. ZUIs, like most visualizations, have strong potential for supporting users in understanding the big picture, identifying trends, patterns and outliers. But they typically are not good at helping users simply get the best answer. So, be sure that some sort of overview-based design is called for in the first place.

Be cautious about using zooming as the primary interface. Zooming navigation can be difficult, and is uncommon. Unless the dataset is fundamentally spatial (such as maps), users probably don't want zooming as their primary interaction with the data. Instead, think about how people search and create data sets - which may then be accessed partially by zooming. I.e., create hybrid systems that combine mechanisms such as faceted search and zooming (e.g., PhotoMesa [11] and Hard Rock site [25]).

Only use ZUIs when a small visual representation of the data is available: Zooming works best when the objects can be recognized when they are zoomed out. So, certain domains are better suited to ZUIs. Photos are good, purely textual documents are bad, and audio recordings are terrible (unless there are associated images). Recent research shows how a small visual snippet may be automatically created from web pages [50].

Don't limit yourself to fixed datasets. People rarely want to interact long with a fixed set of data. But it is often technically difficult to dynamically create and modify large zoomable spaces. Be sure that the user interface does not get blocked or slowed down when new data is added to a zoomable space.

And then there are several more specific suggestions:

Maintain aspect ratio: It is tempting to use semantic zooming to significantly change the visual representation of objects at different sizes. For example, a visual chart may be condensed to a single value or a text document may be replaced by its title when small. However, the small object must have the same aspect ratio – or at least fit in the same box as the full object. Otherwise, nearby objects can end up overlapping when the user zooms out.

Use Meaningful layout: The problem of finding a specific object among a large number of them can be alleviated if the objects are laid out in a semantically meaningful way. If the user can predict where an object will be based on attributes that they are likely to be familiar with, then they can trim the search space. Traditional paper phones books with their alphabetic ordering are a good example of this approach.

Use Consistent layout: Ideally, once an object is placed in space, it shouldn't move. That way, the user can build up an internal map over time of where objects are. Unfortunately, this ideal is often in direct conflict with the previous one of meaningful layout. If objects are to be laid out meaningfully, that implies that the layout changes over time as the data changes and as the organizational mechanism changes. PhotoMesa [11] is an example of a ZUI that went for a meaningful layout but had no consistency at all.

Use Scannable layout: Objects must be positioned in such a manner as to enable the user to visually scan the space to look for an object without missing or repeating objects. Simple 1D lists of objects are easy to scan because the person just runs down the list and the only state that must be kept (i.e., load on short term memory) is the current item and the direction of scanning. In 2D however, for any layout except for a grid, there is the potential for a much heavier load on short term memory.

Use breadth over depth: It is tempting to place objects in space at many different sizes so more information can be discovered by zooming in further – and in fact, many of the earlier ZUI applications did this. However, when objects are placed at many different levels, the user won't know that the small objects exist (unless the space is carefully designed to indicate them), and they can get lost. Further,

even if the user knows they are there, it is typically burdensome to zoom in and out repeatedly. SpaceTree [24] is an example of an application that was designed explicitly with breadth in mind. Even with hundreds of thousands of objects, all of them are at the same size.

Use small datasets: The problems that come with a large number of objects in a zoomable space can be mitigated by not building applications around large datasets. ZUIs appear to work well for users when they have no more than about 100 screens worth of information. However, ZUIs with 1,000 or more screens worth of information become problematic because of the issues raised in the previous section (Why ZUIs are Challenging). This is a fundamental problem since ZUIs are often designed to work for user generated data – which may not easily be limited. So application designers may choose to partition the space into manageable sizes.

Use simple and consistent navigation interaction: People must be able to discover how to navigate the space with no training, and it must be difficult or impossible to get lost in desert fog. Unfortunately, there is significant inconsistency among applications. They vary between single and double clicks to zoom in, whether zooming in zooms in a fixed percentage or makes the target object fill the screen. And mechanisms are all over the map to zoom out – from right click, left click on the background, fixed on screen buttons and popup buttons. And as with zoom in, the actual amount that is zoomed out is inconsistent across applications as well.

CONCLUSION

In retrospect, I believe that we as researchers let ourselves get too caught up in the excitement of something new and coolness of something so engaging. When we couldn't figure out what the killer app in a zoomable environment was, perhaps we shouldn't have been so fast to think that others would figure it out if only we solved the technological problems. Perhaps one or two platforms was a good idea, but were five necessary?

The idea that we should not develop technology without knowing what it is for is dangerous because it is the nature of basic research that you don't always know what a new technology will enable. Transistors and lasers come to mind as technologies that the inventors had no idea what they were good for.

But I believe that in HCI, which is user facing by its nature, it should be much more apparent what the value to users is when developing technology. An example of HCI effort that saw more success only when researchers focused on user needs more than underlying technology is software engineering [39]. We are not developing technology for its own sake, we are developing technology to address user needs, and if we don't start by addressing those needs, we are putting our effort at risk. This is a point that seems too obvious to make except that many of us (especially computer scientists) sometimes miss it.

Further Research

What is the future of ZUIs? Personally, I remain guardedly optimistic. I do not think that people will use ZUIs that are the primary mechanism for managing any significant amount of information as envisioned, for example, by Raskin for managing hospitals [45].

However, as described in this paper, there are a range of potential benefits. For some users and tasks, those benefits will outweigh the costs. The problem, though, is that the precise balance is not well defined. Furthermore, the full range of zoomable designs is surely not fully explored. This leads to the following key areas for further research:

Definitive understanding of benefit: What, specifically are the benefits of zooming transitions, and under what circumstances are those benefits found? Do different users have different capability and preferences when using ZUIs?

Standardized navigation: ZUIs need standardized navigation mechanisms. So many approaches have been used that they need to be catalogued so that it becomes easier to see what are good candidates for standards. And any companies that make more than one ZUI application must ensure that they at least have a company-wide standard.

Design space exploration: Despite all the effort over the years envisioning interaction, design, and application domains for ZUIs, it is a very large space design. There are very likely still new approaches and applications to be discovered. I expect that creative design effort will continue to extend the richness and reach of ZUI applications.

ACKNOWLEDGMENTS

I have worked with many people on ZUIs over the past 17 years since Ken Perlin's first paper on Pad. I can't thank them all, but want to acknowledge the people I worked most closely with on ZUIs, especially in the early days. This includes, of course, Ken Perlin who had the original conceptualization of ZUIs immediately followed by close collaborations on Pad++ with Jim Hollan, Jon Meyer and Allison Druin. I also appreciate the enthusiasm and commercial efforts of Franklin Servan-Schreiber who I worked with when he was at Sony. I worked closely with students Lance Good and Bongwon Suh and staff Jesse Grosjean and Aaron Clamage. Then my later work with Mary Czerwinski and George Robertson at Microsoft and John SanGiovanni, first at Microsoft and then at Zumobi helped me to understand more closely what it really takes to understand an interface, polish a product and make it work for a large audience. And finally, my colleagues at the HCIL (especially Ben Shneiderman and Catherine Plaisant) have been instrumental in being rigorous and ruthless in helping me think about the real costs and benefits of the technologies we build.

REFERENCES

1. Apple iPhone. www.apple.com/iphone
2. Apple OS X Dock. www.apple.com/macosx/what-is-macosx/dock-and-finder.html
3. Bauer, D. S. 2006 The Cognitive Ecology of Dynapad, a Multiscale Workspace for Managing Personal Digital Collections. Doctoral Thesis. UMI Order Number: AAI3222983., University of California at San Diego.
4. Beard, D. B. and Walker, J. Q. 1990. Navigational techniques to improve the display of large two-dimensional spaces. Behav. Info. Tech. 9, 6, 451–466.
5. Bederson, B. B. 1993. Pa3D video. www.youtube.com/watch?v=5JzaEUJ7IbE
6. Bederson, B. B. and Hollan, J. D. 1994. Pad++: a zooming graphical interface for exploring alternate interface physics. In Proc. of the 7th Annual ACM Symposium on User interface Software and Technology UIST 1994. ACM, New York, NY, 17-26. DOI= http://doi.acm.org/10.1145/192426.192435
7. Bederson, B. B., Hollan, J. D., Perlin, K., Meyer, J., Bacon, D., & Furnas, G. W. (1996). Pad++: A Zoomable Graphical Sketchpad for Exploring Alternate Interface Physics. Journal of Visual Languages and Computing, 7, pp. 3-31.
8. Bederson, B. B., Hollan, J. D., Druin, A., Stewart, J., Rogers, D., and Proft, D. 1996. Local tools: an alternative to tool palettes. In Proc. of the 9th Annual ACM Symposium on User interface Software and Technology. UIST 1996. ACM, New York, NY, 169-170. DOI= http://doi.acm.org/10.1145/237091.237116
9. Bederson, B. B., & Meyer, J. (1998). Implementing a Zooming User Interface: Experience Building Pad++. Software: Prac. and Experience, 28(10), pp. 1101-1135.
10. Bederson, B. B., Meyer, J., and Good, L. 2000. Jazz: an extensible zoomable user interface graphics toolkit in Java. In Proc. of the 13th Annual ACM Symposium on User interface Software and Technology. UIST 2000. ACM, New York, NY, 171-180. DOI= http://doi.acm.org/10.1145/354401.354754
11. Bederson, B. B. 2001. PhotoMesa: a zoomable image browser using quantum treemaps and bubblemaps. In Proc. of the ACM Symp. on User interface Software and Technology. UIST 2001. ACM, New York, NY, 71-80. DOI= http://doi.acm.org/10.1145/502348.502359
12. Bederson, B. B., Grosjean, J., and Meyer, J. 2004. Toolkit Design for Interactive Structured Graphics. IEEE Trans. Soft. Eng. 30, 8 (Aug. 2004), 535-546. DOI= http://dx.doi.org/10.1109/TSE.2004.44
13. Bederson, B. B., Clamage, A., Czerwinski, M. P., and Robertson, G. G. 2004. DateLens: A fisheye calendar interface for PDAs. ACM Trans. Comput.-Hum. Interact. 11, 1 (Mar. 2004), 90-119. DOI= http://doi.acm.org/10.1145/972648.972652
14. Boltman, A., Druin, A., Bederson, B., Hourcade, J.P., Stanton, D., O'Malley, C., Cobb, S., Benford, S., Fast, C., Kjellin, M., & Sundblad, Y. (2002). The nature of children's storytelling with and without technology. Proceedings of American Educational Research Association (AERA) Conference, New Orleans.
15. Card, S. K., Moran, T. P., and Newell, A. *The Psychology of Human-Computer Interaction*. Lawrence Earlbaum Associates, 1986.
16. Card, S. K., Robertson, G. G., and Mackinlay, J. D. 1991. The information visualizer, an information workspace. In Proc. of the SIGCHI Conf. on Hum. Fact. in Comp. Sys. CHI 1991. ACM, New York, NY, 181-186. DOI= http://doi.acm.org/10.1145/108844.108874
17. Cockburn, A., Savage, J., and Wallace, A. 2005. Tuning and testing scrolling interfaces that automatically zoom. In Proc. of the SIGCHI Conf. on Human Factors in Comp. Systems. CHI 2005. ACM, New York, NY, 71-80. DOI= http://doi.acm.org/10.1145/1054972.1054983
18. Cockburn, A., Karlson, A., and Bederson, B. B. 2008. A review of overview+detail, zooming, and focus+context interfaces. ACM Comput. Surv. 41, 1 (Dec. 2008), 1-31. DOI= http://doi.acm.org/10.1145/1456650.1456652
19. Combs, T. T. and Bederson, B. B. 1999. Does zooming improve image browsing?. In Proc. of the Fourth ACM Conf. on Dig. Lib. DL '99. ACM, New York, NY, 130-137. DOI= http://doi.acm.org/10.1145/313238.313286
20. Druin, A., Stewart, J., Proft, D., Bederson, B., and Hollan, J. 1997. KidPad: a design collaboration between children, technologists, and educators. In Proc. of the SIGCHI Conf. on Human Factors in Computing Systems. CHI 1997. ACM, New York, NY, 463-470. DOI= http://doi.acm.org/10.1145/258549.258866
21. Furnas, G. W. 1986. Generalized fisheye views. SIGCHI Bull. 17, 4 (Apr. 1986), 16-23. DOI= http://doi.acm.org/10.1145/22339.22342
22. Furnas, G. W. and Bederson, B. B. 1995. Space-scale diagrams: understanding multiscale interfaces. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM Press/Addison-Wesley, New York, NY, 234-241. DOI= http://doi.acm.org/10.1145/223904.223934
23. Good, L. and Bederson, B. B. 2002. Zoomable user interfaces as a medium for slide show presentations. Information Visualization 1, 1 (Mar. 2002), 35-49. DOI= http://dx.doi.org/10.1057/palgrave/ivs/9500004
24. Grosjean, J., Plaisant, C., & Bederson, B. B. (2002) SpaceTree: Design Evolution of a Node Link Tree Browser. Proceedings of Information Visualization Symposium (InfoVis 2002) New York: IEEE, pp. 57-64.
25. Hard Rock Memorabilia. http://memorabilia.hardrock.com

26. Hightower, R. R., Ring, L. T., Helfman, J. I., Bederson, B. B., and Hollan, J. D. 1998. PadPrints: graphical multiscale Web histories. In Proc. of the 11th Annual ACM Symposium on User interface Software and Technology UIST 1998. ACM, New York, NY, 121-122. DOI= http://doi.acm.org/10.1145/288392.288582
27. Hillcrest Labs. www.hillcrestlabs.com
28. Igarashi, T. and Hinckley, K. 2000. Speed-dependent automatic zooming for browsing large documents. In Proc. of the ACM Symp. on User interface Software and Technology. UIST 2000. ACM, New York, NY, 139-148. DOI= http://doi.acm.org/10.1145/354401.354435
29. Hornbæk, K., Bederson, B. B., and Plaisant, C. 2002. Navigation patterns and usability of zoomable user interfaces with and without an overview. ACM Trans. Comput.-Hum. Interact. 9, 4 (Dec. 2002), 362-389. DOI= http://doi.acm.org/10.1145/586081.586086
30. Ishii, H. and Ullmer, B. 1997. Tangible bits: towards seamless interfaces between people, bits and atoms. In Proc. of the SIGCHI Conf. on Human Factors in Computing Systems. CHI '97. ACM, New York, 234-241. DOI= http://doi.acm.org/10.1145/258549.258715
31. Jul, S. and Furnas, G. W. 1998. Critical zones in desert fog: aids to multiscale navigation. In Proceedings of the 11th Annual ACM Symp. on User interface Software and Technology. UIST '98. ACM, New York, NY, 97-106. DOI= http://doi.acm.org/10.1145/288392.288578
32. Karlson, A. K., Bederson, B. B., and SanGiovanni, J. 2005. AppLens and launchTile: two designs for one-handed thumb use on small devices. In Proc. of the SIGCHI Conference on Human Factors in Computing Systems. CHI 2005. ACM, New York, NY, 201-210. DOI= http://doi.acm.org/10.1145/1054972.1055001
33. Klein, C. and Bederson, B. B. 2005. Benefits of animated scrolling. In Ext. Ab. on Human Factors in Comp. Sys. CHI 2005. ACM, New York, 1965-1968. DOI= http://doi.acm.org/10.1145/1056808.1057068
34. Kunkel, P. *Digital Dreams: The Work of the Sony Design Center*. Universe Publishing. 1999.
35. Lichtschlag, L., Karrer, T., and Borchers, J. 2009. Fly: a tool to author planar presentations. In Proc. of the 27th International Conf. on Human Factors in Computing Systems. CHI 2009. ACM, New York, NY, 547-556. DOI= http://doi.acm.org/10.1145/1518701.1518786
36. Microsoft Canvas For OneNote. www.officelabs.com/projects/canvasforonenote/
37. Microsoft pptPlex. www.officelabs.com/projects/pptPlex/
38. Microsoft Seadragon. www.seadragon.com
39. Myers, B. and Ko, A. J., "The Past, Present and Future of Programming in HCI". Human-Computer Interaction Consortium (HCIC 2009), Winter Park, CO. Feb. 2009.
40. North, C. and Shneiderman, B. 2000. Snap-together visualization: evaluating coordination usage and construction. Int. J. Hum.-Comp. Stud., 53, 5, 715–739.
41. Perlin, K. and Fox, D. 1993. Pad: an alternative approach to the computer interface. In Proc. of the 20th Annual Conf. on Computer Graphics and interactive Techniques. SIGGRAPH 1993. ACM, New York, NY, 57-64. DOI= http://doi.acm.org/10.1145/166117.166125
42. Piccolo2D. www.piccolo2d.org
43. Prezi. www.prezi.com
44. Rao, R. and Card, S. K. 1994. The table lens: merging graphical and symbolic representations in an interactive focus + context visualization for tabular information. In Proc. of the SIGCHI Conference on Human Factors in Computing Systems: Celebrating interdependence. CHI 1994. ACM, New York, NY, 318-322. DOI= http://doi.acm.org/10.1145/191666.191776
45. Raskin, J. *The Humane Interface*. Addison-Wesley Professional, 2000.
46. Rekimoto, J. 2002. SmartSkin: an infrastructure for freehand manipulation on interactive surfaces. In Proc. of the SIGCHI Conf. on Human Factors in Computing Systems. CHI 2002. ACM, New York, 113-120. DOI= http://doi.acm.org/10.1145/503376.503397
47. Robertson, G., van Dantzich, M., Robbins, D., Czerwinski, M., Hinckley, K., Risden, K., Thiel, D., and Gorokhovsky, V. 2000. The Task Gallery: a 3D window manager. Proc. of the SIGCHI Conf. on Human Factors in Comp. Sys. CHI 2000. ACM, New York, 494-501. DOI= http://doi.acm.org/10.1145/332040.332482
48. Schematic. www.schematic.com
49. Sun, L. and Guimbretière, F. 2005. Flipper: a new method of digital document navigation. In CHI 2005 Extended Abstracts on Human Factors in Computing Systems. CHI 2005. ACM, New York, NY, 2001-2004. DOI= http://doi.acm.org/10.1145/1056808.1057077
50. Teevan, J., Cutrell, E., Fisher, D., Drucker, S. M., Ramos, G., André, P., and Hu, C. 2009. Visual snippets: summarizing web pages for search and revisitation. In Int'l Conf. on Human Factors in Computing Systems. CHI 2009. ACM, New York, NY, 2023-2032. DOI=http://doi.acm.org/10.1145/1518701.1519008
51. van Wijk, J. and Nuij, W. A Model for Smooth Viewing and Navigation of Large 2D Information Spaces. IEEE Transactions on Visualization and Computer Graphics, 10 (4). 2004, 447-458.
52. Ware, C. *Information Visualization, 2nd Ed.* Morgan Kaufmann, 2004.
53. Zoomorama. www.zoomorama.com
54. Zumobi. www.zumobi.com

Year	Name	Description	Layout Flexibility	Mult-level	Navigation Mechanism For Zooming
1997	KidPad	Uses "local tools" [8] to enable children to create stories [20].	Unconstrained. Drawings, images, text positioned manually.	Yes	Hyperlinks for path. Zoom in/out modes. Click to fly in/out.
1998	PadPrints	Creates a visual hierarchical map of web pages visited [26].	Dynamically generated tree-based node-link diagram	No	Right click-and-hold flies in/out.
2001	CounterPoint	A plug-in to PowerPoint that enables presentation authors to lay out slides in zoomable space [23].	Contains PowerPoint slides. Small number of layout algorithms wth manual override.	Yes	Click on slide for next. Right click-and-hold flies in/out.
2001	PhotoMesa	A personal photo browser, desktop and mobile [11].	Contains photos in a dynamically generated group of grids using a treemap algorithm.	No	Left click zooms in. Right click zooms out.
2002	Seadragon	Client/server high perf. ZUI for images over a network (web and mobile) [38].	Contains photos laid out by a small number of fixed layout algorithms.	No	Click to zoom in. On-screen button zooms out.
2002	Spacetree	Hierarchy exploration tool [24].	Dynamically generated tree-based node-link diagram	No	Right click-and-hold flies in/out.
2006	Dynapad	Supports organization of personal collections [3].	Dynamic, very flexible.	No	Multi-touch
2007	Apple iPhone	Application launcher for Apple's mobile phone [1].	Contains icons laid out in a fixed grid.	No	Tap to zoom in. Physical button to zoom out.
2008	Microsoft pptPlex	A plug-in to PowerPoint that enables presentation authors to lay out slides in zoomable space [37].	Contains PowerPoint slides. Small number of layout algorithms with manual override.	No, but with manual override	Click on slide for next. Click to zoom in. Esc to zoom out. Scroll wheel flies in/out.
2008	Zoomorama	Client/server to enable high performance zooming of images over a network [53].	Dynamically generated grid containing images.	No	Pop-up on-screen buttons for zoom in/out.
2008	Prezi	Website for creating and making free-form presentations [43].	Unconstrained. Drawings, images, movies, text positioned manually.	Yes	Tab for next. Scroll wheel, on-screen buttons and mouse/keyboard combo to fly in/out.
2009	Zumobi* ZoomCanvas	Launcher for a small number of related content areas on high end mobile devices [54].	Fixed hand-tuned layout turned to content.	No	Tap to zoom in, on-screen button to zoom out.
2009	Microsoft Canvas for OneNote	Plug-in for Microsoft OneNote shows visual overview of all a user's notes [36].	Dynamically generated grid of grids containing images of notes from Microsoft OneNote.	No, but with manual override	Click on note to zoom in. Esc, right click, or bgnd click to zoom out.
2009	Schematic.com	Example of custom website with fixed layout of fixed content [48].	Fixed hand-tuned layout tuned to content.	No	Click on slide to zoom in. Click on background to zoom out.

Table 1. Some ZUI Applications. *Note that I, the author, am a cofounder of Zumobi, Inc.