WebBubbles

A Multi-Touch Browsing Experience

Alexander Blessing
Stanford University
ableSS@stanford.edu

Arda Kara
Stanford University
ardakara@stanford.edu

Yu-Ta Lu
Stanford University
luyota@stanford.edu

ABSTRACT

We describe WebBubbles, a zooming user interface for the iPad that visualizes links on a web page. WebBubbles allows users to select links of interest on a web page and loads them in a new cluster. Users can pan and zoom into the newly generated clusters and thus browse multiple pages more effectively on the iPad. Our initial informal user studies show that users respond very well to the novel interface, and can more quickly browse and explore multiple links.

Author Keywords

Mobile; Browser; Zooming; Panning; Clustering; Glance; Multiple links

ACM Classification Keywords

H.5.2 Information Interfaces and Presentation: User Interfaces—*Graphical user interfaces (GUI)*

INTRODUCTION

Every day we are faced with web pages that contain multiple links we would like to follow. A very common user behavior is to open the links in new tabs first, and then go through them one by one. This web browsing pattern is known as *hub-and-spoke browsing* [1], and is supported on desktop browsers with the ability to open a link in a new tab with only one click accompanied by a keyboard button press. The currently ubiquitous tab-based system has some shortcomings. If more than a couple links are to be visited, clicking on them one by one can get strenuous. Also, once opened, tabs retain no information about what page they were outlinked from. This results in users not remembering what the page contained in a tab was relating to, or why they even opened it in the first place.

Furthermore, the process gets more cumbersome on touch screen platforms such as Apple's iOS operating system. The Safari browser on iOS mobile devices makes its users long-touch a link and select the "Open in New Page" option to be able to open a link in a new tab (as seen in Figure 1). A competent user takes two touches and at least a few seconds

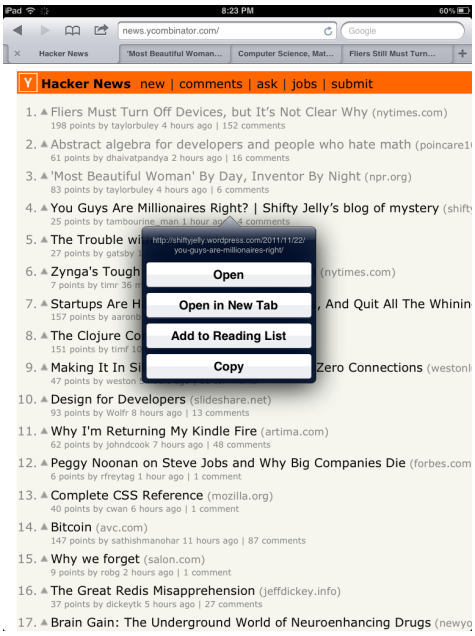

Figure 1. The current process of opening a link in a new tab on the iPad

for each link they would like to follow, which makes hub-and-spoke browsing on the iPad very impractical.

Once the links are opened in new tabs, browsing through them is also very cumbersome. The non-tablet versions of the software automatically switches to the newly opened tab, which makes it very difficult to open a list of links in their own tabs. Changing between tabs also takes at least a touch (to get into tab changing mode), a couple swipes (to get the right tab at the center of the screen) and another touch (to view it). We see that taking the same metaphors that enable *hub-and-spoke browsing* on desktop browsers and carrying them over to mobile doesn't just brings their shortcomings, but also creates new and worse ones.

With WebBubbles we propose a new web browsing paradigm that utilizes the natural panning and zooming gestures available on touch screens to make *hub-and-spoke browsing* more practical on mobile devices. Our goal was to make selection of multiple links, browsing the newly opened pages and seeing the relationship between the pages as easy as possible. We also strived to design an interface that was enjoyable to use and worked with users' current assumptions about how browsers behave.

Submitted for review to CS448B '11.

RELATED WORK

Web related visualization has been an active field of research starting with the 1990s. Among the work that is relevant to our project are a web browser for small terminals that uses text-reduction and focus+context visualization [2], a system for visualizing the web with a navigational view builder [3], a framework to visualize user browsing paths on the web called WebQuilt [4] and a web browser that can move temporally called Zoetrope [5].

Zooming user interfaces (ZUI) first started getting attention with the Pad project in the early 1990s [6]. Browser applications of ZUIs came with the Pad++ project in the late 1990s [7]. However most of the useful ZUI characteristics of Pad++ were abandoned in modern browsers. More surprisingly, these features were also left out when mobile browsing on smaller screens became popular [8], since the then prevalent desktop browsing conventions were directly carried over. More work has been done in the area in the 2000s to discover ways of making the larger size of webpages work better with the smaller screens of mobile devices using panning and zooming [9] and visual summarization methods [10]. Nevertheless, these zooming interfaces only considered intra-document zooming. Inter-document zooming and panning for mobile made an appearance in 2008 with Firefox Mobile Concept Video in 2008 [11], but fell short in incorporating additional useful features that utilize this paradigm. Combination of clustering and zooming can be seen in an extended abstract describing HishiMochi [12] which works with a static set of documents and utilizes clustering for topical relation.

WebBubbles draws on a wide variety of past research; however, to our knowledge our solution is novel in both its context and its application. The multi-touch screens of today make panning and zooming feel very natural through swiping and pinching fingers on the screen. Combining this with the increasing memory and network bandwidth on mobile devices make it a practical and enjoyable solution to use.

Figure 2. Upper picture shows user drawing a rectangular area on the bottom left side of the CNN home page. Lower picture shows the generated preview of those links

METHODS

We designed WebBubbles, an iPad application that lets users browse web pages and visualize links that they want to explore. Our application consists of two major parts. The *link selection mode* allows users to browse a web page as they normally would, and then enables them to effectively select a set of links they want to visualize. Next, the *visualization mode* uses this set of links to generate a cluster of web pages representing those links. This gives the user a quick visual glance and preview of the web page, allowing him to zoom in to the generated web pages of interest, interact with them further and generate new visualizations from there. Figure 2 illustrates a typical WebBubbles use case.

In following sections, we will give a more detailed account of the algorithms and techniques we employed in both the selection and visualization mode.

Link selection mode

When the application is first started, the user can specify a web page he would like to browse. At that point, he can interact with the web page as he normally would. When he wants to visualize a set of links, he enters the link selection phase and specifies the links of interest.
We faced several challenges in designing the link selection phase. First, there needs to be a quick, easy and unambiguous way to enter the selection phase. Second, the user must be able to flexibly and conveniently select a set of links, that may or may not be in adjacent places on the web page.

Our solution to the first problem entails putting a button on the top left corner of the application, allowing the user to enter the link selection mode. Once the link selection mode has been entered, the user can no longer interact with the web page (e.g. by panning or touching on links); instead, he can now tap on specific links he wants to visualize. We give the user a visual feedback about which link he tapped on by drawing a small rectangle at the relevant position. By

allowing the user to tap on individual links, it is possible to visualize links that are located on completely different positions on the screen, which fulfills our requirement of flexibility. We also allow the user to tap and drag a bigger grey rectangular area (as shown in Figure 2). The application will then visualize all links that are located under that area. This feature addresses our design goal of offering convenience, since many times the links of interest are located adjacent to each other.

Visualization mode

Once the user selects a set of links, he or she can enter the visualization mode by touching a button on the top right corner of the application. This will zoom out of the web page, create a new cluster with web pages corresponding to the links, connect it to its source web page with an edge and zoom in to the newly generated cluster.

By embedding the canvas view in a scroll view, we allow the user to pan and zoom around the new cluster, and indeed the whole screen. That is, the user can zoom in to some of the newly generated pages for more detail at a different zoom level, or zoom out and pan back to the original page he came from. By drawing a line from the original page to the newly generated cluster, we keep a strong visual relationship between those pages. This enables the user to keep track of his browsing history, and provides him with useful and relevant context.

By flying through the web views in the generated cluster, the user can pinpoint the page he is most interested in and choose to further explore it. Upon a double tap on the relevant page, the application smoothly zooms in on that page, making it full screen and allowing the user to interact with it as he would in any common web browser. From here on, he can either zoom back out again, or enter the link selection mode, generate a new cluster, zoom in to one of the new pages, generate another cluster, and so on. By using a line as visual representation of the original and generated clusters, the application allows the user to effectively step back in his browsing history and not get overwhelmed.

Indeed, one concern was that the user might try to visualize a too large number of web pages. This could be problematic in two ways. First, it would clutter a cluster with too much information and images, also putting a strain on the computing and memory resources of the device. More importantly, however, the primary benefit of visualizing a set of pages - namely the quick visual glance, allowing the user to recognize pages of interest - would diminish, as the individual pages would get smaller and the user would be too overwhelmed by the abundance of information. For that reason, a design choice that we made was to limit the maximum size of a cluster to be 9 pages. If the user selects a set of more than 9 links, only the first 9 links will be visualized.

Another important design choice was the layout of the clusters, i.e where on screen to put a newly generated cluster. There has been a lot of work in the area of free space modeling. Bernard and Jacquenet [13] give an algorithm for placing rectangular areas in free space without overlapping. Similarly, the Pad++ browser [14] uses a basic layout function that assigns a degree-of-interest value to each node in the tree based on its distance from the focus page.
However, our premises differ significantly from the existing applications. First, we give the user the ability to zoom in and zoom out of web pages, fully interact with newly generated ones and create new clusters on the fly. Moreover, the underlying device, the iPad, provides a completely enhanced user experience of panning and zooming. That is, while previous GUI interfaces were not designed to support zooming and panning in a natural way (and had to make design choices that accounted for these limitations), zooming and panning is an essential part of the ”iPad experience”. For that reason, the precise layout of the cluster is less important. Our design choice was to use a linear left-to-right flow of the clusters: a new cluster is placed to the right of the original page/cluster it was generated from. If two (or more) clusters are generated from the same cluster, then the two (or more) clusters are still both displayed to the right of the original cluster, but displaced vertically from each other.

Finally, a technique we used to efficiently deal with a large number of web pages was to show an image of the web page instead of the web page itself once the page was fully loaded. That is, once the user starts the visualization mode, we load the actual web pages in a new web view. This will load the entire DOM and fill up the memory resources on the device very quickly. Our workaround is to catch a notification that's triggered once a web page has finished loading, at which point we take a snapshot of the web view and replace it with that image. This enables us to visualize a large number of web pages, and keep a smooth zooming and panning interface. Only when the user zooms in to a page by tapping on it does the application reload the web view, allowing the user to interact with it fully. During small user trials, we noticed that the transition from the snapshot image back to the actual web view is actually negligibly small and does not impact the user experience.

RESULTS

Case Study: Visualization Result of Pinterest

Here we analyze the visualization results of the webpages http://pinterest.com.

Pinterest is a social photo blog that allows the user to organize the picture he is interested in and share with other people. As shown in Figure 3, the website is designed to provide smaller version of photos on an overview page where the user can browse multiple of them quickly. The smaller version of these photos are also linked to other pages that have their higher resolution ones along with user comments. Therefore, a typical user behavior of this page is that the user checks out the overview page first, and if he sees anything of interest he either opens up the link in a new tab to check later, or directly taps on the link and then hist the back button after checking the larger version of photo. As we mentioned before, both of these ways are not ideal when there are many photos the user is interested in because first, the iPad tab can only be opened after long pressing the link

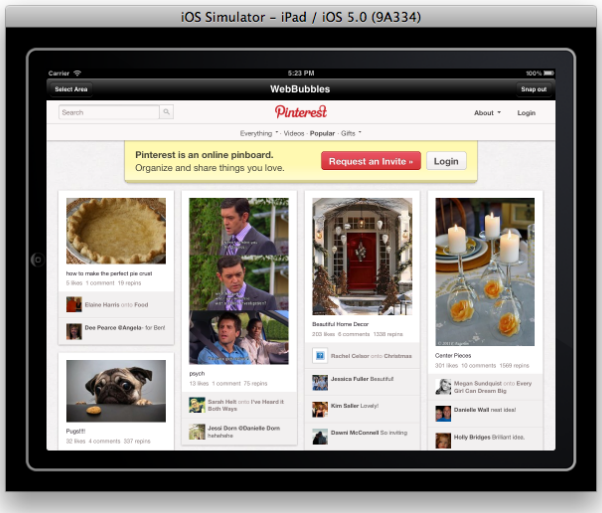

Figure 3. Pinterest Webpage

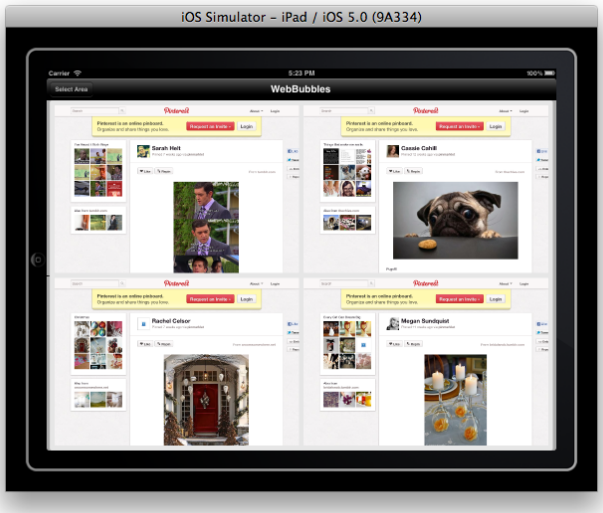

Figure 4. Pinterest Cluster of Visualized Links from The First Webpage

which is not very efficient, and second, oscillating back and forth between pages disrupts the browsing flow.

With WebBubbles, multiple links selection is provided for avoiding the common two types of inefficiency. After selecting several links with dragging and tapping, the user gets the result in Figure 4. The cluster of links provides the user with an overview of all the links the user is interested in. Since the main focus of the pages are photos, the user can quickly recognize the pages. Furthermore, he can then zoom in to the size of a single web page for a larger version of photo or pan around to navigate different pages.

If after such interaction, the user finds out some more interesting posts, he can explore more by repeating the above process. Usually, these photo blogs allows the user to follow certain pages or authors. Thus, when the user sees an interesting page, he might want to know the browsing history and understand how he reaches the page so that he can trace back to the source of his visit to learn more information from that page. This use case is supported by our visualization of web browsing history as shown in Figure 5.

User Test Results

We informally user tested WebBubbles on around 10 users. We first explained the idea, demonstrated the application to them, and then let them interact with it by themselves. All users like the novel way of browsing the web and were looking forward to seeing the application published on the app store.

We also observed the user behavior when they are experimenting with the application. While the users were able to smoothly operate the application, we found some minor interface usability issues. First, the line between a webpage and its associated cluster might need to be designed more carefully. Currently it is a straight line drawn on top of the screen. The problem of such implementation is that the line in some sense blocks the visual content of other webpages, and users sometimes found it annoying when having to see the lines when they were navigating the pages. Second, there is not a way to remove the pages once they are visualized. There should be some error recovering mechanism that is used when the user accidentally visualizes the page he is not interested in. Finally, the original design of snapping into a page by double tapping on it is unexpected. The users expect to single tap to snap into the page. These feedback provides great direction for our future work.

DISCUSSION

Our WebBubbles browser aims to tackle the problem that the user needs to perform repetitive and inefficient interaction with the traditional tab based browser on iPad as they are browsing the page. The lack of keyboard input and supports of gestures on an iPad are important factors that affect a user's web browsing experience. WebBubbles includes these factors into its design which leads to the following advantages over a tab based browser on an iPad.

Efficiently Opening Multiple Links of A Webpage

Since there is no keyboard input on an iPad, the user usually needs to long press on the link of a webpage to open the link in a new tab in a tab based browser. This restriction is minor when it comes to opening a single link in a webpage, but will result in great inefficiency when the user attempts to open several links while keeping the current contents displayed. For example, a link generally takes 1.5 second of long pressing and clicking on the "open in new tab" button in the popup menu to open, and opening 10 links will take at least 15 seconds. Our design solves this problem by enabling the user to easily select an area or single tap on multiple links to open all selected links at once. This design allows the user to open consecutive links (multiple links laid out next to each other) as well as individual links (links scat-

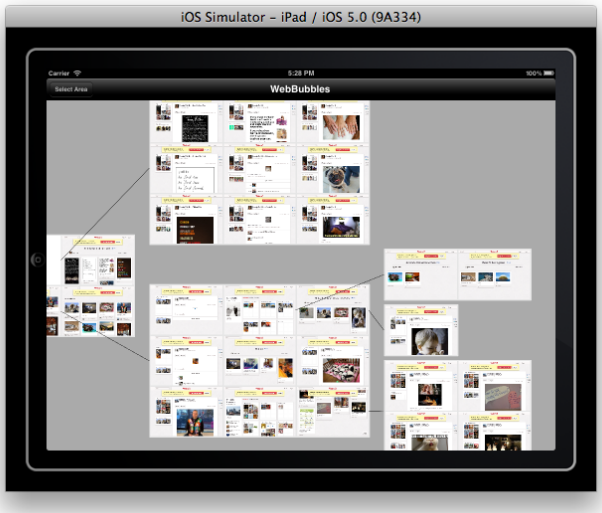

Figure 5. Visualization Result of Pinterest Browsing History

tered on the page, seperate from each other). It reduces the time and effort needed to open multiple links.

Viewing Contents in Different Links at A Glance

Browser tabs are designed to allow the user to review the title of the tab and click on it to quickly jump to the intended content. Therefore the title plays an important role of helping the user to identify which tab to jump to. That is, it serves as the content cue for the page. However, as the number of tabs decrease, the visible part of the title shrinks, and it becomes harder for the user to read the complete title to understand the content in the tab quickly and thus decreases the usability of the browser. This is especially true on a smaller device such as the iPad where the width of the screen is very limited.

Our solution provides the user with an easier way to identify webpages using a visual way. We use a grid of screenshots of pages to work as the visual cue of the contents of multiple webpages. Since the grid having the page information is now displayed in a full browser window rather than the tabs in a single row, a better glance of pages is thus provided. Furthermore, the epxerience is enhanced by the gestures supported on the iPad. First, we allow the user to zoom in and out of the grid combined with panning on the grid to adjust the level of detail and the specific part the user wants to see. Second, the user can jump to a webpage by tapping on it, and as a result the browser snaps into that page automatically and the user can now interact with the page itself. To leave the page, a snap out button can be tapped to go back to the grid view. This interaction technique helps provide more webpage content cue and information.

Note that here we emphasize the power of visual cue of webpages. However we are not suggesting that the title of a webpage will be less useful. In fact, we believe the combination of the visual cue and text will be more powerful than either technique alone.

Easier Navigation Between Pages

In a tab based browser, as the number of tabs the user opens increases, the context of each page is lost. One common scenario is that the user is browsing a webpage with many interesting links. He starts by opening up all the links he is interested in in new tabs. Then he starts to browse some of these pages and finds out that there are even more interesting links in them, and thus he again opens these interesting links in new tabs. Sooner or later he finds the browser is overwhelmed with all the tabs without knowing why these tabs are opened and what webpages lead to these tabs.

WebBubbles solves this problem by clustering pages from the same source together and providing a hierarchical view of clusters. Staring from one page, the user selects multiple links to open. The newly opened links will be shown in a new cluster. Furthermore, we visualize the link between the original page and the new cluster to give the user the context of the cluster so that he knows where these clusters are coming from. As the number of clusters increases, our layout system will rearrange the location of each cluster to prevent one from overlapping with each other, and thus can create a very clear tree structure of the web browsing graph. This graph shows the way the user's browsing history. In addition, with the panning and zooming gestures the user can easily navigate through the tree which leads to less confusion as he is browsing the web.

FUTURE WORK

Based upon the very positive feedback that we have received from users, we plan to polish WebBubbles and publish it to the App Store soon. For future work, there are several valuable extensions that can be made.

One feature of zooming user interfaces that has not yet been fully implemented in WebBubbles is having different levels of detail at different levels of zoom. This idea is very common with zooming user interfaces, most prominently Google Maps [15], where the application shows more granular detail at a finer zoom level. This is related to the idea of detail on demand [16], a very popular visualization technique.
It is not immediately obvious how to apply this to WebBubbles. In particular, an important design choice is what information to abstract at the zoomed out level. For example, one might imagine just showing a picture - the favicon or the biggest picture on the web page - or maybe try to provide a one-sentence summary of the page (such as the title in the header). However, it is not even clear whether this is preferable to just showing the entire web page, even at a highly zoomed out level.

A more obvious feature to implement is the ability to close certain web pages that are not of interest. Currently, there is no means to hide a generated web page within a cluster, and so even erroneously generated pages stay on the view. Our informal user studies show that users like to be able to quickly close irrelevant pages.

Third, even though it was a concrete design choice to layout the clusters in a linear left-to-right fashion, some feedback we received voiced the wish to be able to reposition clusters. It appears beneficial to give users the ability to easily drag-and-drop clusters to a new position. The touch-and-pan interface that the iPad provides seems very suitable for this interaction.

Finally, we are interested in exporting and analyzing the graphs that get generated by users. On the one hand, we believe that this might serve as useful information to the users themselves, enlightening them about their browsing behavior. Conversely, the generated graphs might affect a users' browsing behaviors. For example, users might try to optimize for a beautifully layed out graph, and adjust their browsing behavior accordingly. We believe that this is a question worth exploring.

REFERENCES

1. L. Tauscher and S. Greenberg, "Revisitation Patterns in World Wide Web Navigation", CHI 97, 1997.
2. S. Bjrk, et al., "WEST: a Web browser for small terminals", Proceedings of the 12th annual ACM symposium on user interface software and technology (UIST '99), 1999.
3. S. Mukherjea, J.D. Foley, "Visualizing the World-Wide Web with the Navigational View Builder", Proceedings of the Third International World-Wide Web Conference, Volume 27, Issue 6, pp 1075-1087, 1995.
4. J.I. Hong, J.A. Landay, "WebQuilt: A Framework for Capturing and Visualizing the Web Experience", Proceedings of the 10th international conference on World Wide Web (WWW '01), 2001.
5. E. Adar, et al., "Zoetrope: Interacting with the Ephemeral Web", UIST '08, 2008.
6. K. Perlin and D. Fox, "Pad - An Alternative Approach to the Computer Interface", ACM SIGGRAPH 93, 1993.
7. B.B. Bederson, et al., "Pad++: A Zooming Web Browser", SPIE Multimedia Computing and Networking, Volume 2667, pp 269-271, 1996.
8. A. Kivi, "Mobile Browsing", Topical Evolution Paths of Mobile Services, 2007.
9. P. Baudisch, et al., "Collapse-to-zoom: viewing web pages on small screen devices by interactively removing irrelevant content", Proceedings of the 17th annual ACM symposium on user interface software and technology (UIST '04), 2004.
10. C. Gutwin and C. Fedak, "Interacting with big interfaces on small screens: a comparison of fisheye, zoom, and panning techniques", Proceedings of Graphics Interface (GI '04), 2004.
11. Firefox Mobile Concept. http://mozillalabs.com/blog/2008/06/firefox-mobile-concept-video. Retrieved Dec 2011.
12. M. Toyoda, E. Shibayama, "HishiMochi: a zooming browser for hierarchically clustered documents", Extended Abstracts on Human factors in computing systems (CHI EA '00), 2000.
13. Bernard and Jacquenet. Free space modeling for placing rectangles without overlapping. Journal of Universal Computer Science, vol. 3, no. 6 (1997).
14. Bederson, B.B., Hollan, J.D., Stewart, J., Rogers, D., Vick, D., Ring, L.T., Grose, E., Forsythe, C. A Zooming Web Browser. Human Factors in Web Development, Eds. Ratner, Grose, and Forsythe, Lawrence Erlbaum Assoc., pp 255-266, 1998.
15. Google Maps. http://maps.google.com. Retrieved Dec 2011.
16. Ball: Robert Ball, Details on Demand vs. High Resolution, 2004.