SEARCH ENGINE RESULT PAGE WITH VISUAL CONTEXT AND ALREADY RENDERED SNIPPETS

Kazutaka Maruyama
Information Technology Center, The University of Tokyo, 2-11-16, Yayoi, Bunkyo, Tokyo, Japan
kazutaka@acm.org

Masato Igeta, Minoru Terada
Dept. of Information and Commun. Eng., The Univ. of Electro-Communications, 1-5-1, Chofugaoka, Chofu, Tokyo, Japan
igeta@ice.uec.ac.jp, terada@ice.uec.ac.jp

Keywords: Search Engine Result Page, Snippets, User Interface

Abstract: Search engine result pages, aka *SERP*, provide page titles, URLs, and text based snippets of the searched pages. These help users to decide to click or not each searched page. Text based snippets, however, are different from the real page rendered by web browsers, after a user clicks one of the searched pages, the user may see an unexpected one. Even if the page is the expected one, the user cannot find immediately where the information they want to know is described in the page.
In this paper, we propose *visual patch*, which is provided as a graphical snippet in SERP and is rendered by web browsers already. Visual patch is a visual context based snippet, not a thumbnail of the whole web page in a smaller size, and is trimmed around the occurrence of the query keyword in the same size as the original page rendered by web browsers. Users can see the layouts around the keyword and read the concrete text just before clicking the searched pages. Thus visual patch in SERP shows not web pages which may include the information users want to know, but the information they want to know or not.

1 INTRODUCTION

We use web daily to retrieve, check up on, and exchange the information. Search engine result pages, aka SERP, include (1) titles, (2) URLs, and (3) text based snippets (Fig. 1). Users judge whether each searched page includes the information they want to know by the three kinds of the information about each result. Snippets provide the abstract of the page or a part of the sentence around the keyword and help users to decide whether the page includes the information they want to know. It is known that snippets are important and useful for users to choose a URL to be clicked(Takaku et al., 2009).

Text based snippets are different from the real pages rendered by web browsers, therefore, a page may not include the expected information even if a user clicks the result. For example:

- Snippets do not provide the context of the sentence around the keyword.
- The keyword is included not in the body but in the advertisement.
- HTML/CSS decorations takes no effect to snippets.

Context in this paper means not only the flow of sentences but also the typesetting such as `li`, `h2`, font sizes, or the color of the text designated by HTML/CSS. Usual search engines display text based snippets at a restricted area in their SERP. In other hands, a page has multiple occurrences of the keyword and each occurrence may be related to a different topic. Text based snippets essentially have their limitations of the ability of providing the appropriate information to the search engine users. In addition, users essentially want to know not web pages which may have the information they want to know, but the information itself. Putting a whole page into a short text of snippet does not fill this gap.

In this paper, we propose visual context based snippets, *visual patch*. Visual patch shows the fragment of the web page around the keyword rendered by web browsers. This enables us to provide the concrete information in a web page instead of the whole web page. A pilot user experiment is also included. The rest of this paper organized as follows: section 2

Figure 1: Search engine result page with text based snippets.

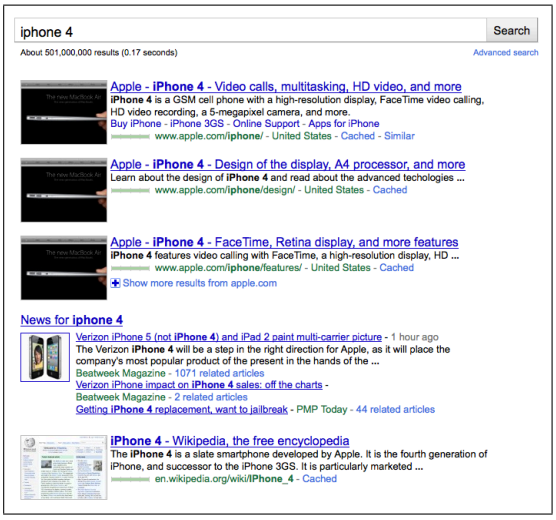

Figure 2: Giving "iPhone 4" to Google with SearchPreview provides the thumbnails including a new MacBook Air.

compares our proposal with some related works, section 3 describes the details of visual patch and its use case, section 4 shows how to produce SERP with visual patch, section 5 evaluates our proposal qualitatively through user experiments, section 6 discusses some issues of visual patch, then section 7 describes the conclusions.

2 RELATED WORK

There are several works which provide the rendered web pages. SearchPreview(Ackroyd, 2010), a Firefox add-on, provides thumbnails of the whole web pages of each result in SERP. The thumbnail is the top of the site, for example, the top page of www.example.com for www.example.com/a/b.html (fig. 2). Each thumbnail is cached in the server of SearchPreview and the add-on combines the images with the usual SERP. Thumbnails of whole web pages are useful to know the kind of the site or to decide whether it is already known or not. The images, however, shrink and the users cannot read the contents of the web pages. Most of the top page of the web sites do not represent each page of the site.

Teevan et al.(Teevan et al., 2009) propose the method of providing a combined snippet of the characteristic words or images of the page (fig. 3). For tab browsers, Liu et al.(Liu and Tajima, 2010) provides the thumbnails of the whole web pages and the characteristic images of the pages in the same size as the original in order to help users to remember the contents of the tabs (fig. 4). Woodruff et al.(Woodruff et al., 2001; Suh et al., 2002) propose the method of providing the overlays of the keywords in addition to the thumbnails of the whole web pages. These proposals are effective for users to choose a search result from the viewpoint of the category of the keywords or to be reminded of the already known web pages. However it is not suitable to find out the information never seen or to understand the context of the text. Our proposal, visual patch, enables users to read the description of the body and know the typesetting around the keyword.

3 VISUAL PATCH

In this section, we describe our prior work and then show the improved SERP produced by our pilot system.

3.1 Search within a page

We already proposed the system which provides the visual search result within a page(Igeta et al., 2009). Fig. 5 shows the search result of a certain web page of Wikipedia, which describes a certain university in Japan. The search keyword within the page is the name of a certain institution of the university. The name appears in (1) the table of contents, (2) the list of the institutions of the university, and (3) the index of the section described about the institution and the three small windows show each occurrence. If a user want to know the description of the institution, the

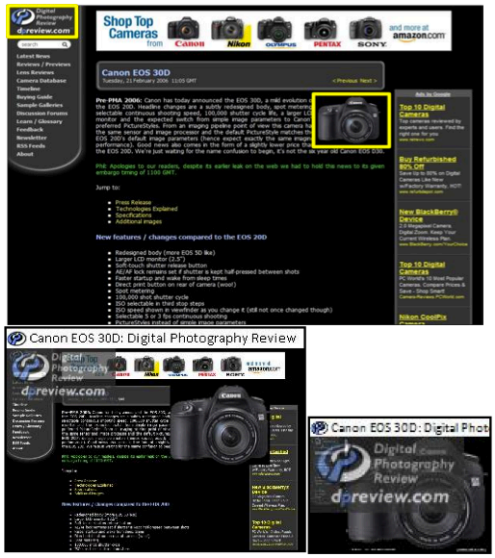

Figure 3: Visual Snippets(Teevan et al., 2009) provides the image at bottom right from the one at top half.

Figure 4: WildThumb(Liu and Tajima, 2010) provides a few images in their original size included in the pages. The thumbnails are located at both sides of the browser.

person can choose the third one to be lead there. If another want to know other institutions of the university, the second window helps the person. In this paper, we call these windows *visual patches* (VP).

These would be distinguished from each other in terms of document object model of HTML. But both the first and the second are a part of the list items and the difference between them is almost meaningless. To introduce a heuristics that a numbered list item with anchor tag may be an index is effective only to the limited situation. In other hands, displaying the search result as the rendered image in the same size with the decoration designated by HTML/CSS in that page helps us to judge whether it is the information we want to know or not.

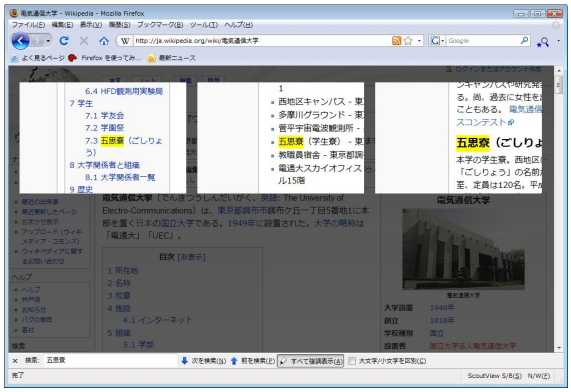

Figure 5: Visual Patch in a certain page(Igeta et al., 2009).

3.2 SERP with Visual Patch

Listing the occurrences of a keyword within a web page is also useful when we are searching web pages by search engines. VP enables search engines to provide the appropriate information about the given query in SERP rather than the list of the web pages. Fig. 6 shows an example of SERP with VP produced by our pilot system. The current implementation requires two queries to produce a SERP: (1) the query for searching web pages and (2) the query for searching itself within the pages. We call the former "search engine query" and the latter "VP query". Using "obama approval rating" as search engine query and "average" as VP query causes the result shown in fig. 6 [1].

In the case of this example, the left and center VPs in the row (1) and the left one in (2) are the information a user want to know. The right in (1) and the left in (2) shows the list of the index, so these are the unexpected one for the user. We can also see that the right in (2) and the left in (3) do not show the Obama's general approval rating directly.

The color of the frame of each VP is assigned by the web page in which the VP is included. The left and center in (1) come from the different web pages, but have the same contents; the one has the copy of the other. We can find that these contents are in Wikipedia in terms of the design of the page. Please note that the snippets of SERP are replaced with VPs and VP shows not the summary of the web page but the information in the web page itself.

[1] The caption of "(1)" to "(3)" in the figure are added by us for the explanation.

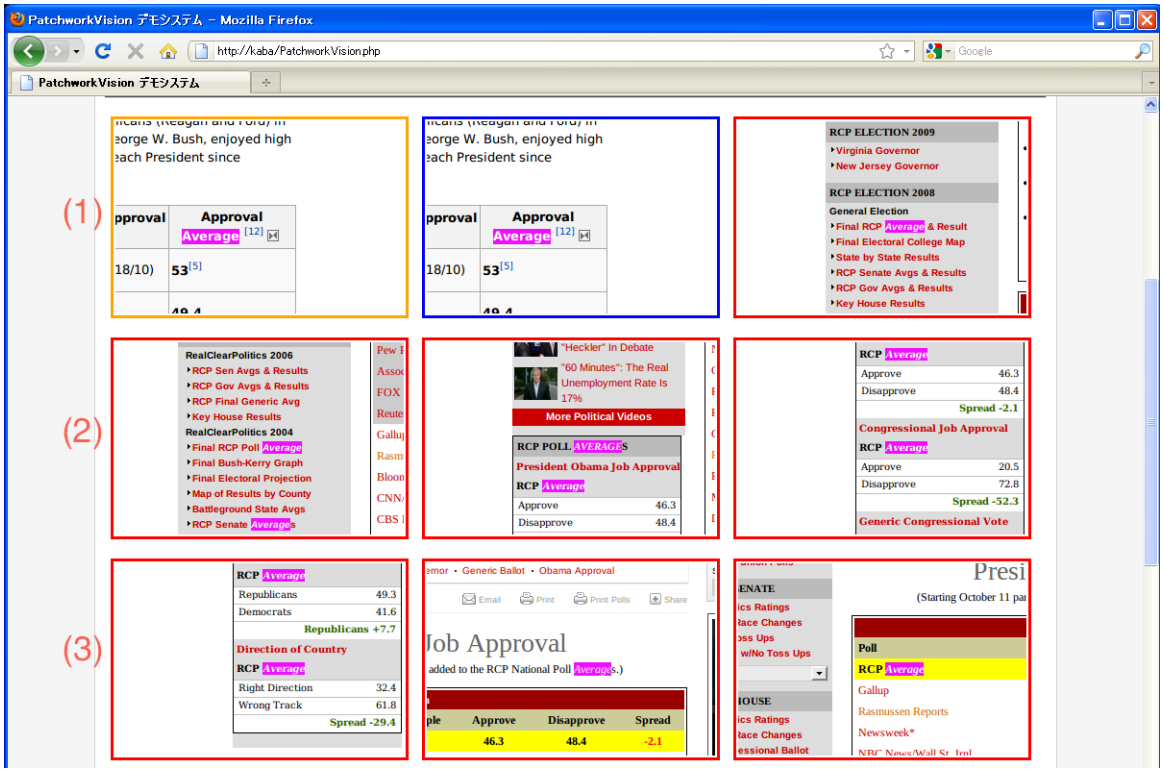

Figure 6: SERP with visual patch. Search engine query is “obama approval rating” and VP query is “average”.

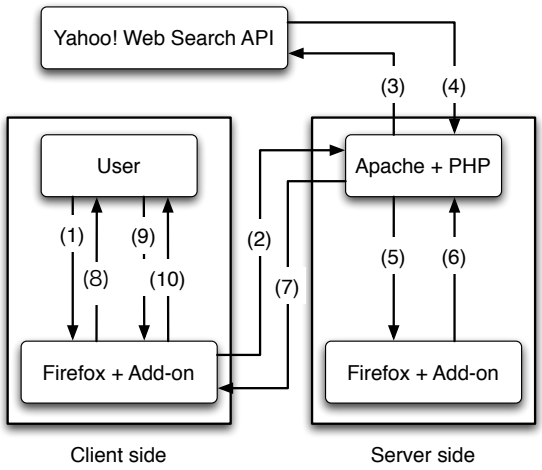

Figure 7: Overview of the pilot system.

4 IMPLEMENTATION

The pilot system producing the SERP with VP shown in fig. 6 is composed of three components (fig. 7).

- Client side component which interacts with users.
- Server side one which produces VPs by rendering web pages and provides SERP.
- Yahoo! search API which returns the list of URLs as the result of search engine query.

Both the client and server side components include Firefox web browser with our add-on. The server side add-on automatically produces VPs, and the client side one automatically opens and scrolls the web page when a user clicks a VP. Ten steps in the figure are as follows:

1. A user inputs both search engine query and VP query to the web form of our system.
2. Firefox sends the queries to the CGI programs written in PHP on the server.
3. The CGI program sends the search engine query to Yahoo! search API.
4. Yahoo! returns the list of URLs as the result of the search back to the server.
5. The CGI program invokes Firefox to open each URL in the list and tells VP query to the server side add-on.
6. Firefox and the add-on on the server render the web pages, highlight the keywords of VP query in

each page, capture all the occurrences of the keyword to produce VPs, and then sends VPs back to the CGI.

7. The CGI returns a web page of SERP including VPs back to the client side Firefox.
8. Firefox shows the SERP with VPs as the result of the submit of the queries in step 2.
9. The user clicks a certain VP.
10. Firefox opens and renders the web page including the VP and the add-on automatically scrolls the page to the occurrence point of the keyword of VP query.

The pilot system limits the number of web pages rendered to the top ten URLs of the result from Yahoo! in step 5, in order to reduce the response time from the input by the user to the output of the result. The server side Firefox renders the ten pages to produce VPs from them. The current server has Intel Core i7 2.8GHz and 4GB memory and the response time is at least several seconds, at most ninety seconds. The server side component uses the full function web browser with GUI to render web pages and is on a single machine. Just a rendering engine could be used for producing VPs instead of web browser. Using multiple machines to parallelize the production of VPs and caching the rendered pages are required to improve the response time.

5 USER EXPERIMENTS

We evaluated our pilot system by user qualitative experiments through three tasks for nine students. In the first task, we designate the goal of the task, search engine query, and VP query. In the second, we do only the goal, such as "to search the article about the institute of the university". In the third, we do nothing and the students decide their goal by themselves. We received the following comments:

- It is good to see the contents of web pages in visual context.
- The response time is too long.
- Effective if the information I want to know is relatively clear.
- The choice of VP queries is difficult.
- It is easier to search the already known pages.
- Good size of VPs.
- The inappropriate queries produced the bad results.

These comments show that the system is useful when the information to be searched is relatively concrete, or when the page to be searched is already known in terms of its design. The inappropriate queries produce worse result, because the number of VPs depends on the number of the occurrences of the VP query rather than the number of the pages returned from Yahoo! search API.

6 DISCUSSION

6.1 Visualization vs Space

VPs provide richer information than text based snippets in compensation for the loss of space in the users' display. In the desktop environment, it is popular to use a large display such as over 24 inches or to use multiple displays simultaneously, and VPs obviously have an advantage over text based snippets.

In the mobile environment, a display of mobile devices is usually not enough and a few VPs may occupy its area. Text based snippets also show only two or three results and require users to scroll the display in order to show more results. At that point, VPs and text based snippets are even. When the user transfers SERP via 3G network instead of Wi-Fi, text based snippets may be preferable.

6.2 Contents of VP

There are positive comments to the contents of VPs. Especially, some works described in section 2 emphasise only the keywords and reduce the size of the contents of the web pages. As a result, we cannot read the sentences around the keywords. Our proposal has an advantage over them.

The range and size of VPs should be still discussed. In the current system, the keyword of VP query is placed at the center of the VP and both the number of pixels and the aspect ratio are determined by experience. Actually, how to trim the web page around the keyword would depend on the layouts of the page. For example like blogs, the left and right columns include the links to other dates or the profile of its author, and the center one hold the body of the blog and the comments from others. When the keyword is placed at the left edge of the center column, the keyword should be placed at the left in the VP rather than just the center of the VP. This problem could be resolved by identifying the layout of the keyword in terms of DOM of HTML.

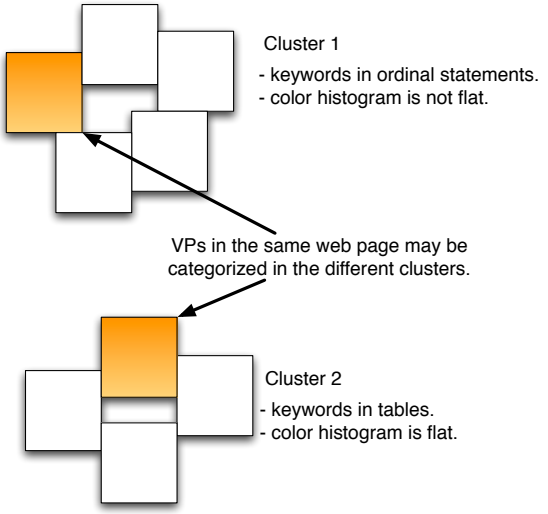

Figure 8: VPs could be categorized in some groups.

6.3 Confidence of VP

The fifth point of the comments of the user experiments designates not only the effectiveness of VPs but also the need to deal with the unknown web pages. When the users search web pages without any expectations of result, they use the titles or URLs as clues to choose web pages in SERP. For example, if a user finds out the specifications of a certain commercial product, the person relies on the maker's web site. The combination of VPs and the thumbnails of the whole web pages may be required.

6.4 Clustering VPs

Although the current system produces few VPs, when the system in future produces much more, we must discuss how to provide many VPs to users in SERP. Clustering VPs may help this problem. Fig. 8 shows an example of separating VPs in two groups. For example, group 1 includes VPs of which the keywords are in the usual sentences and group 2 includes VPs of which the keywords are in the tables. In another example, VPs are categorized based on their color histogram. This grouping could represent the visual impression of the design of the page.

The figure shows one more important thing. It is acceptable that two different VPs in the same web page are categorized in the different groups. As the information users want to know is not a web page but just the information itself included in the web page, the information on web should be provided to users by VPs instead of web pages.

7 CONCLUSIONS

The usual SERP provides only the links to the web pages which include the query. It just shows the web pages which may include the information the user want to know. As SERP with VP provides the concrete and readable information included web pages, the user can choose the appropriate result in SERP. We continue to consider some issues discussed in section 6 and improve the response time of the system.

REFERENCES

Ackroyd, E. (2010). *SearchPreview :: Add-ons for firefox*. Retrieved February 7, 2011, from https://addons.mozilla.org/ja/firefox/addon/189/.

Igeta, M., Terada, M., and Maruyama, K. (2009). Scoutview:scrolling support interface for web page. In *IPSJ SIG Technical Report*, volume 2009-HCI-133. Retrieved from http://www.bookpark.ne.jp/ipsj/ (In Japanese).

Liu, S. and Tajima, K. (2010). WildThumb: A Web Browser Supporting Efficient Task Management on Wide Displays. In *Proceedings of the 14th International Conference on Intelligent User Interfaces (IUI2010)*, pages 159–168.

Suh, B., Woodruff, A., Rosenholtz, R., and Glass, A. (2002). Popout Prism: Adding perceptual Principles to Overview+Detail Document Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computer Systems (CHI2002)*, pages 251–258.

Takaku, M., Egusa, Y., Saito, H., Miwa, M., and Kando, N. (2009). Analysis of User Eye Movements during Viewing of Search Engine Results Pages in Web Information Seeking Tasks. *Japan Society of Information and Knowledge Journal*, 19(2):224–235. (In Japanese).

Teevan, J., Cutrell, E., Fisher, D., Drucker, S. M., Ramos, G., Andre, P., and Hu, C. (2009). Visual Snippets: Summarizing Web Pages for Search and Revisitation. In *Proceedings of the 27th International Conference on Human Factors in Computer Systems (CHI2009)*, pages 2023–2032.

Woodruff, A., Faulring, A., Rosenholtz, R., Morrison, J., and Pirolli, P. (2001). Using Thumbnails to Search the Web. In *Proceedings of the SIGCHI Conference on Human Factors in Computer Systems (CHI2001)*, pages 198–205.