EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling (2005)
Takeo Igarashi, Joaquim A. Jorge (Editors)

Retrieving Web Page Layouts using Sketches to Support Example-based Web Design

Yasunari Hashimoto[1] Takeo Igarashi[1,2]

[1]The University of Tokyo [2]JST PRESTO

Abstract

It is very difficult for inexperienced users to design good-looking web page layouts. Even with WYSIWYG editors, it is still difficult to obtain desired results because one must combine complicated HTML specific structures. Another reason is that it is inherently difficult for those who have limited artistic talents to design good layout from scratch.

We address these problems by providing an environment where one can design web page layouts using existing well-designed web pages on the net as examples. The system basically works as a search system. The user sketches a simple layout as a query and the system returns web pages that have similar appearance. By referring to the example pages, the user can learn the basic principles of HTML design, obtain inspiration for better design, and also can directly use the professionally designed layouts by replacing the contents.

This paper describes the user interface and the implementation of the system in detail. It also reports the result of an empirical user study. The users found the system very useful and the layouts designed using the system was scored highly by evaluators.

Introduction

The design of web pages has become more and more important as an increasing number of people rely on them as daily information resources. The quality of web page is strongly affected by its appearance [IHS01], but it is usually difficult especially for a novice user to design a good-looking layout. The process of trial and error is widely considered to be necessary for good design [GL85]. However, most web design tools are focusing on the creation of high-fidelity prototypes and are not appropriate for an iterative design process. There is a sketch-based system for the early stages of web page design [LNHL00], but it mainly focuses on the design of hyperlink structures and does not support the process of encoding layouts in HTML format.

This paper introduces a system that assists inexperienced users in web page layout design process by providing examples. It basically works as a web page search system. It takes a simple layout sketch as a query and returns a collection of existing web pages on the net whose appearance is similar to the query (Figure 1). In other words, this system turns a simple sketched layout into actual layouts defined by HTML tags. We expect that this system helps inexperienced users in web page layout design process in three different ways.

First, it helps inexperienced users to understand what kinds of layouts are possible in the tag-based HTML framework. It is difficult for inexperienced users who are not familiar with the behavior of HTML tags to imagine what kinds of layouts are possible. However, our system visualizes how their final page design might look like, which can help the user to learn web page design principles.

Second, it helps users to explore multiple design possibilities starting from their vague images in mind. Hand-drawn sketches are helpful in the very early design stages [LNHL00]. However, even after settling to use a particular sketch, there are still multiple possibilities in the process of turning the sketch into an actual HTML document. Our system allows the user to quickly compare multiple design possibilities in their close-to-final appearance without hand-encoding individual HTML documents and this can facilitate an iterative trial-and-error design cycle.

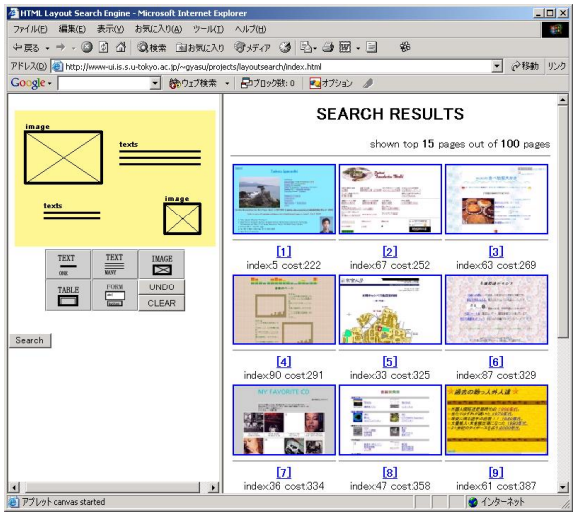

Figure 1: *The user interface of the system. The user specifies a simple layout as a query (left) and the system returns a collection of example web pages that have similar appearance (right).*

Finally, it helps users to construct an actual HTML document using tags, after finalizing the iterative design process. Since our search system returns not only the final web page visuals as seen in browsers but also the source html files with tags, the user can simply replace the content of the preferred HTML document with the intended content keeping the layout information.

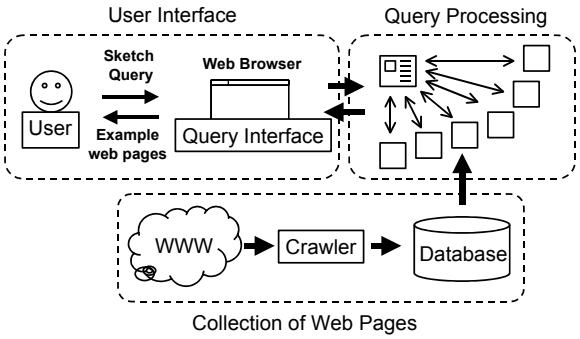

Figure 2: *The overview of the system.*

Our system mainly consists of three components: a sketch-based query interface running on a browser, a query processing engine that finds the pages in a database that match the query, and a crawler that collects web pages and builds the database (Figure 2). As shown in Figure 1, the user generates a query on the left panel of the web browser as a simple sketch and the system shows the search result on the right panel as thumbnails. The system computes the matching scores between the input sketch and example web pages stored in the database and returns the pages with the highest scores. Our special purpose crawler collects web pages on the net and stores them in the database with appropriate index information in an offline process.

1. Related Work

There are popular tools for creating web pages such as GoLive, WebSphere, HomePageBuilder, and Dreamweaver. These systems use WYSIWYG interfaces where the user can edit the document visually without explicitly typing HTML tags. This approach is useful when the user has a specific layout already in mind and is familiar with the web page layout principles as to what kinds of layouts are possible. However, WYSIWYG interfaces are not very helpful in the early stages of design because the editing process in a WYSIWYG interface does not support the quick exploration of multiple possibilities.

DENIM [LNHL00] is a sketch-based design tool for early stage of web design. Their user study showed the rapid sketch interface is effective for making a design. However, DENIM is designed for professional designers who can easily derive more detailed web pages from their rough sketches. Inexperienced users are not sure how their sketch might appear when converted into a HTML document. Our system helps them in the process of turning ambiguous layout designs in mind into actual web pages.

WebStyler [HGLS98] generates an actual HTML file from a simple sketch. It can help users to quickly obtain an html page corresponding to the input sketch. Bouillon et al. also use a graphical drawing tool for specifying the layout of web pages in their system for porting a web site from one platform to another [BVC04]. However, the resulting design might be not be very sophisticated because it relies completely on the limited design skill of the users. Using our system, inexperienced users can learn a lot from professionally designed web pages on the net and they can explore multiple possibilities quickly, which is important for good design.

Our approach works as a search engine for web page layouts. The most widely used search engines use keywords [HC01] while some search engines use various search methods including natural language [Ask] and SQL-based queries [FLM98]. There are also search systems for finding visual contents such as images [Fli95][SM99][SC96] and 3D models [MHKF03]. However, none of them helps the users to find web pages with a desired layout.

Fonseca presented a search system for vector drawings [Fon04]. The user sketches approximate shape as a query and the system returns similar drawings as a result. While their system is designed for general drawings, our system is specialized for web layouts. They consider the geometrical similarity of individual elements, but we only take element type and their relative size and position into account.

Cullen et al. introduced a layout-based search system for PDF documents [CHH97]. It is specialized for strongly formatted documents for print and is not applicable for HTML documents. To be specific, they focus on the detailed appearance of textual contents such as fonts and density of texts. However, web pages have a limited variety of text appearances and the general location of visual objects is far more important.

2. User Interface

The users can find web pages by specifying desired layout through a web browser. Sketch takes an important role for good design [GL85], so we designed sketch-based query interface based on observations how users sketch web pages.

2.1 Pilot User Study

Prior to the implementation of the system, we conducted an informal user study to learn how people sketch given web pages. We need this information to design the user interface of the system. Six university students participated in the study. They all had experience in creating their own web pages, but they were not professional web designers. We presented 10 web pages that include visible HTML consisting of text, image, table, form, list, and horizontal line. The participants were asked to freely draw corresponding query sketches for these 10 web pages

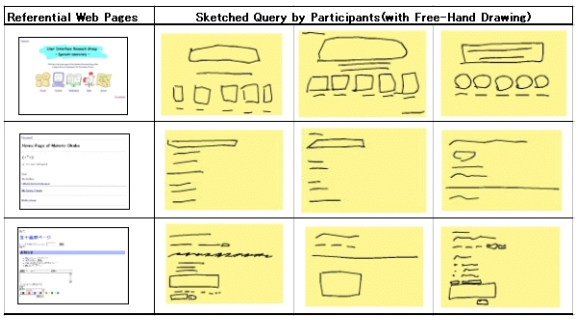

Figure 3: *An example of the result in the pilot user study*

Figure 3 shows some results of the study. While there is a large diversity among the resulting sketches, we made the following valuable observations.

1. Almost all visible objects on a web page are represented as a simple line or a rectangle. Texts with default font size are usually drawn as lines and texts with larger fonts, images, tables and input forms are drawn as rectangles.

2. Even though there are many objects on a web page, users usually don't take care of these objects exactly. They draw only some objects considered to be important for them. Dots at the beginning of list items and long horizontal lines do not appear in sketches even though they are visually distinctive on web pages. These elements seem to be judged as inessential for layout design.

3. Many participants seemed stressful to draw many identical elements repeatedly (especially lines in a text). So it is considered useful to be able to specify multiple text lines easily.

We designed user interface for specifying a query based on these observations.

2.2 Design Detail

Figure 4 shows the panel for specifying a query. It consists of a drawing canvas and buttons for specifying an object type. First, the object type is selected with buttons at the bottom. They are corresponding to HTML elements of one-line text, multi-line text, image, table, and form respectively. A multi-line text is added based on our observation 3 in the preliminary user study. We did not include lists and horizontal lines according to observation 2. The user can freely place objects on the canvas as in standard object-oriented drawing editors. Because of observation 1, a text is displayed as a line and other objects as rectangles on the canvas.

It might be better to allow the user to draw freeform strokes for a query and make the system recognize them as HTML elements [CGFJ02][PFJ00]. As an initial experiment, we choose to ask the user to explicitly specify object types in this current implementation to avoid recognition errors. It is our future work to support freeform sketching as a query and to investigate its effectiveness.

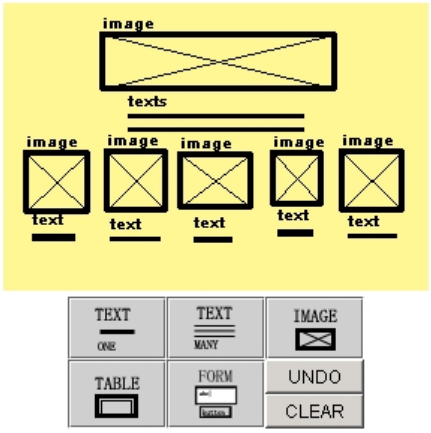

Figure 4: *A snapshot of the panel for drawing query sketch*

The current implementation does not support varying canvas size, so the user cannot exactly represent vertically or horizontally long pages (current size of a canvas is fixed 284 x 192(pixel)). It is our future work to support varying canvas size, but we observed that this is not a real problem

because the default view (that we can see first when we visit the page) has the profound visual impact and it is usually shown in a standard size on the screen [Sch].

The system starts the search when the user pushes the "search" button and shows the search result in the right panel as thumbnails (Figure 1). These thumbnails also work as a link to the original page, so the user can quickly check the visual appearance of the page in full size and also see its source HTML code.

3. Collection of Web pages

Prior to individual search, the system collects web pages by running our own web crawler over the World Wide Web and stores them in a database in the preprocessing.

3.1 Filtering Unnecessary Pages

Ordinary web crawling systems are usually encouraged to download as many files as possible. However, our crawler avoids web pages that are not useful for layout design such as pages with very similar layouts and extremely simple pages. Our heuristic to remove similar pages is to restrict maximum number (10 in our experiment) of downloads from same URL path because these pages tend to have similar appearances. The pages with a few objects (less than 4 in our current experiment) are removed because they are not useful as examples.

3.2 Crawling Process

Figure 5 shows the overview of the crawling process. The crawler is written in Java and searches for files by following links in the pages. Our crawler downloads not only HTML files but also image files contained in those HTML files. We maintain a hash table of visited URLs to avoid visiting pages more than once [PSM93].

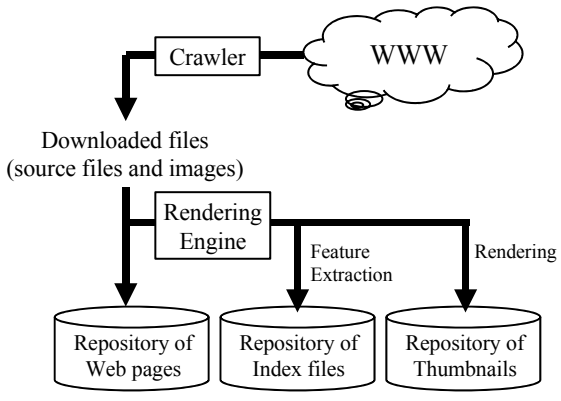

Figure 5: *The overview of the crawling process*

Every downloaded HTML file goes through several preprocessing steps. The crawler internally runs a rendering process and creates thumbnails from HTML source files for the display of the search result. At the same time, the system generates an index file that includes spatial information of all objects in each HTML document.

3.3 Rendering Process

A HTML rendering engine is attached to the crawler as an internal process. It parses the downloaded files and computes the position of objects to create thumbnails and index files. A thumbnail is used to display the search result. An index file is created to accelerate the search because it takes time if it renders the all source files in a database each time.

An index file is a plain text file that contains each visual object's tag type, location of the center point, and area. Images and forms are processed as independent visual objects. A collection of letters in a single line of text and a table with multiple cells inside are also processed as independent visual objects. Visual objects are associated with their tag type (text, image, table, or form).

The appearance of a web page is affected by window size. For example, the long text line is folded according the window size. We fix the window size to match the canvas size in the user interface. Our current rendering engine does not handle web pages containing frames and plug-ins such as Java applet and Macromedia Flash. These web pages are removed during the crawling process.

We run our web crawler on a 2.00 GHz Pentium IV Machine with 264 MB memory and obtained 925 web pages. We spent approximately a day for the collection in our current implementation without performance optimization.

4. Query Processing

The query processing engine compares the user drawn sketch and the example pages in the database and returns pages that best match the query. The heart of the process is the computation of the similarity score between the query and example pages in the database. Given the scores for each example page in the database, the system simply collects the pages with the highest scores and returns them as the search result. We first briefly review existing algorithms for computing similarity between images. We then describe our algorithm and an empirical evaluation of the algorithm.

4.1 Background

Our algorithm is based on similarity search algorithms for content-based image retrieval [MCL97][RHC99]. According to [MCL97], these are basically divided into four types (Figure 6). The differences of those algorithms lie in

which kind of image features they focus, such as color, texture, shape and spatial relations. The algorithms that focus spatial relations are further divided into those use 2D strings [CSY87][LH90][PO96] and attributed relational graphs (ARGs) [Pet02].

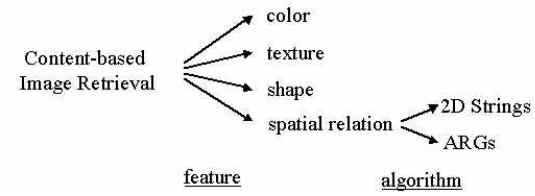

Figure 6: *Simple classification of the feature and algorithm used in the content-based image retrieval*

2D strings are fast but not very accurate while ARGs return better solutions with its expensive cost of time and space. Considering the balance of response time and accuracy, we implement the matching algorithm based on an ARGs algorithm called Hungarian method [Pet02]. It calculates cost between all combinations of objects and finds the best match.

We use size and position of objects in addition to tag type for cost calculation. As we mentioned in Section 4.3, this information is contained in the index files.

4.2 Computation of Matching Cost

The computation of the similarity is based on the spatial relationship between the visual objects on the screen. We currently focus on the difference in size and the distance between corresponding visual objects. For example, the computation between a user query and an example page in Figure 7 proceeds as follows:

1) The system retrieves the tag type, position and size of each visual object in the example page using the index file.

2) We define a cost function f for a pair of visual objects. We designed the function so that a well matching pair of visual objects returns small cost. The definition is as follows:

$$f(obj, obj') = \text{distance}(obj.center, obj'.center) + w \times \sqrt{|obj.area - obj'.area|}$$

$$if\ (obj.tag \neq obj'.tag)\quad f(obj, obj') = c + f(obj, obj')$$

where obj and obj' are visual objects in the user query and the example page. If the tag types are the same, the cost is computed as a weighted sum of a distance between the central points and a square root of the area difference. If tag types between two objects are different, a constant value c is added to the basic cost. Currently, the value of w is 1.0 and c is 600.

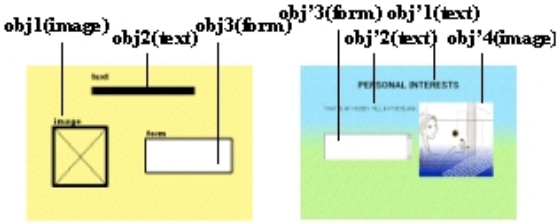

Figure 7: *Example query (left) and web page (right).*

3) The similarity cost between the query and the example page is calculated as follows.

$$\text{total cost} = \sum_{(i,j)\in U} f(obj_i, obj'_j)$$

$$\text{extra cost} = \sum_{i \notin U} \sqrt{\text{obj}_i.area} + \sum_{j \notin U} \sqrt{\text{obj}'_j.area}$$

$$\text{similarity cost} = \frac{\text{total cost} + \text{extra cost}}{|U|}$$

The cost f is first computed for all possible pairs of objects in the user query and the web page. Then the system searches for the optimal set of pairs U such that each visual object in the query is associated with one in the web page and the total cost becomes the minimum. We use a greedy algorithm (sort the pairs according to the cost and add pairs one by one if both nodes are available) for the search. Hungarian method tries to find an optimal answer but we found that our greedy approach is fast and gives reasonable result. In addition to the total cost associated with U, the system also computes extra cost associated with the visual objects that are not contained in U (white object in Figure 8) by taking the square root of their area. The similarity cost is computed by dividing the sum of the total cost and the extra cost by the number of pairs |U|.

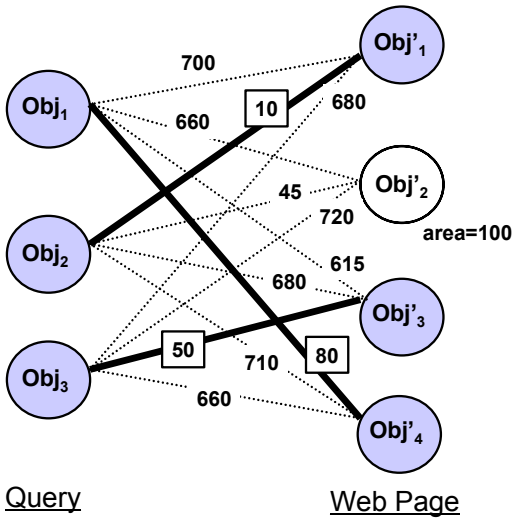

Figure 8: *Minimum cost pairs among objects in the query and web page shown in Figure 7.*

The similarity cost between user query and web page is ((10+80+50)+ $\sqrt{100}$)/3 = 50 in this example.

4) Finally, the system computes similarity cost between the query and the all example pages in the database. The result of the search is presented to the user in the ascending order of this cost.

4.3 Evaluation of the Algorithm

We conducted a simple evaluation user study to see to what extent our algorithm can find appropriate web pages for a given sketch query. Our approach is to compare the output of our algorithm with the presumably "right" results that are selected manually by the test subjects.

Dataset

We manually selected 100 web pages from a web page collection gathered by own crawler for the evaluation. The overall usability of the proposed system is tested later in Section 6 using larger collection of web pages. However, our focus here is the matching algorithm and we preferred controlled setup in this study.

Participants

Ten university students participated in our study. They include the six participants in the pilot study for designing the user interface. All have experience in creating their own web pages, but they are not professional web designers.

Procedure

Each participant is first requested to draw a query sketch for the five representative web pages (Figure 9) in the dataset (these pages are called the original pages). These samples are chosen beforehand so that they represent some of typical layout patterns. Then, for each sketch query they have drawn, the participant is asked to choose four web pages from the dataset (these pages are called the similar pages) as desirable search result for the query. The original page and similar pages are ordered according to the similarity to the query sketch and they are used as the "correct" search result for the query sketch (the original page is always the first).

Figure 9: *The five original pages presented to the participants.*

Result

Given the queries and the corresponding "correct" answers including five original pages and twenty similar pages obtained per participant by the above procedure, we now can measure the accuracy of our matching algorithm. We applied the matching algorithm to the query sketches and observed how the original pages and similar pages are distributed in the search result. Figure 10 shows some of the results and Figure 11 shows the distribution.

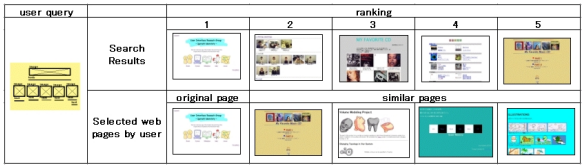

Figure 10: *A typical example of the results. The user draws the query based on the original page, and also manually selects similar pages. The original page is successfully retrieved by the system, but the some similar pages are missing.*

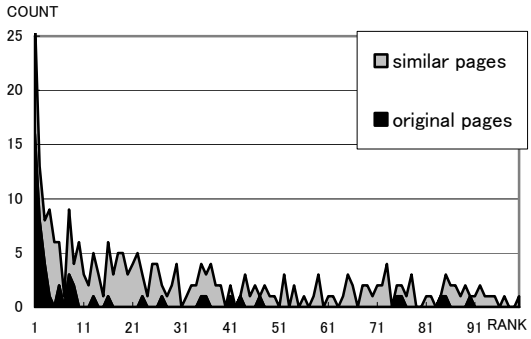

Figure 11: *The distribution of the original pages and similar pages in the search results. Each page is allotted to its ranking in the search result and this histogram shows the number of pages allotted to each ranking. Ideally, the original pages are ranked 1st and the similar pages are ranked within 5th.*

The result shows the system successfully gave the highest scores to the original pages that are given to the participants (90 percent of the original pages are ranked within the top 10). This means that if the user has a specific layout in mind and carefully draws the query sketch, the system can successfully find the right answer. As for the similar pages, the 50 percent of the similar pages chosen by the participants are ranked within the top 20. This is a reasonable result, but some similar pages are ranked very low which means that there is room for improvement.

In terms of the response time, we obtain search result 2.3 sec for 100 pages and 11.8 sec for 925 pages with fif-

teen objects in a query. We use a machine with Pentium 4 2.79 GHz with 1GB memory.

5. User Study

We conducted a user study to evaluate the effectiveness of the proposed system. We asked subjects to design web pages with and without the system and compared the quality of the two.

5.1 Set Up for the Study

Participants

Seven university students participated in the study (Table 1). One of them has never designed a web page (novice user). Two have designed web pages more than four times and they are used to web page design (skilled users). The remaining four subjects have less experience (intermediate users). The half of the intermediate and skilled users usually refers other web pages when they start to design a new web page.

Table 1: *The profile of the participants. Half of them reported that they refer other pages when designing new pages.*

Participant	Experience	Refer other pages
1	*Novice*	No
2	*Intermediate*	No
3	*Intermediate*	No
4	*Intermediate*	Yes
5	*Intermediate*	Yes
6	*Skilled*	No
7	*Skilled*	Yes

Apparatus

The machine that we used for the evaluation is TOSHIBA P4 (Pentium 4 2.79GHz with 1.0 GB memory) laptop running Windows XP SP1. We run a web server (Apache 1.3.29 for Win 32) internally on the machine. The system had 925 web pages in a database and the users interacted with the system using a web browser (Internet Explorer 6 with Sun JDK 1.4.1). The participants used IBM Home Page Builder 8 for the design.

Procedure

Each participant was requested to create two web pages with same contents (images and texts) using a commercial WYSISYG editor (IBM HomePageBuilder 8). They first designed one web page from scratch without the system. They then designed another by referring example web pages found using our system. After finding a favorite page, the user opened it in the editor and pasted the images and texts onto the example page. They were encouraged to make as good-looking web pages as possible. To help motivate the participants to create the best design, we offered 3000 yen to the best design.

We are aware that this within-subjects condition may cause some learning effect that favors our system. However, it is difficult to apply between-subjects condition in this design task because of large individual differences. We believe that this user study can provide enough information for the informal validation of our approach.

At the beginning of the test we instructed participants how to use the editor for approximately thirty minutes and how to use our system for approximately five minutes. The participants chose one of the three types of page contents: self-introduction, photo album, and diary. We provided image files and a text file as the target contents of the web page. The participants are asked to design a page using these contents. The size, color and position of those contents are free. Small modification of those contents was also permitted for the good layout. The time for creating each web page was twenty minutes.

5.2 Results

Informal Observations

All participants especially novice and intermediate users had trouble in editing and checking the layout design many times when they design a page from scratch. They said that they had no final layout image in mind at the beginning and tried to turn the obscure image into an actual design through trial-and-error process. However, that process was tiring even on a graphical web page editor. Only two skilled users performed this process more fluently than others.

When the participants used our system, we made several valuable observations. The users first tried various types of layout on a canvas in the user interface prior to the edit. This quick trial-and-error process helps the user to build a concrete image in mind for the final design. Additionally, the users were able to construct web pages rapidly by modifying the example pages found in the search.

Another observation was that the participants learned general color/layout principles of web pages by looking at the search results. One user who made self-introduction page using black color as a background color remarked instructive aspect of our system.

At first, I used black as a background color for self-introduction page. However, I found that few self-introduction pages use black background on the net after checking the thumbnails. So, I chose lighter background when creating the second page.

The interesting effect that we observed was that several users were not easily satisfied with their result when they tried to design the second web page using the proposed

system. The search result reveals that the user's work leaves room for improvement. One user remarked about this effect.

When I made a web page from scratch, I was satisfied with my work because I thought there was no room for improvements. However, the search results showed many possibilities to improve my web page design. So, at the second time, I couldn't satisfy myself within the given time slot because I had many ideas to make my page look better.

The participants sometimes had trouble in finding good referential pages during the search. This is either because the search result does not match the user query or the search result does not contain attractive designs even the results matches the query. We discuss this issue and possible future improvements in the next section.

Quality Assessment

Given the set of web pages designed with and without the system, we can now measure the effectiveness of the proposed system. We recruited another ten persons as evaluators and asked them to determine which one of the two looks better. They do not know how these pages are created. Figure 12 shows some web pages designed by the participants and Figure 13 shows the result of the assessment by the evaluators.

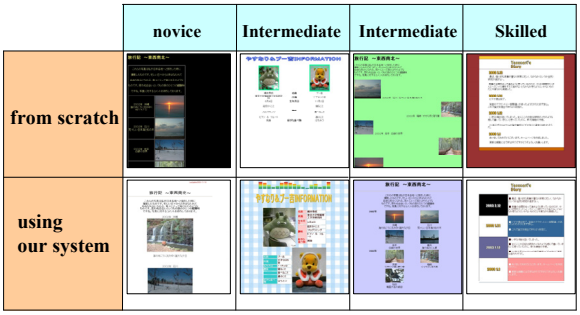

Figure 12: *The resulting web pages designed by the subjects. In general, the pages designed using the proposed system (bottom) are more sophisticated than those designed without the system (top).*

The result shows that the evaluators found that web pages created using the proposed system are better in general. Only one skilled user performed better without the system. He is a skilled designer and has an ability to design and implement good-looking layout from scratch. He mentioned that the quality of the example pages in the system was poor and they hindered his creative imagination. On the other hand, the other skilled user performed much better using the system. He has never referred other web pages when he designs web pages, but he mentioned that he found it is very useful to see example pages.

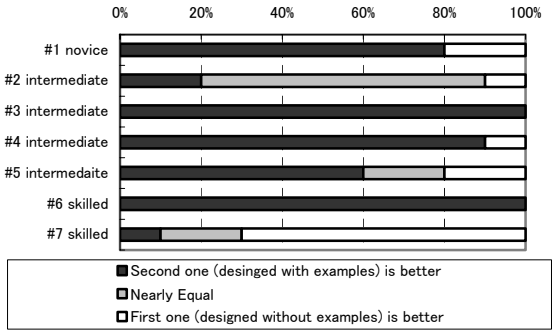

Figure 13: *The quality assessment of the resulting design. Many evaluators found that the pages designed using the proposed system are better than those designed without the system.*

Three participants reported that they would like to use the proposed system when designing web page in the future. Other three liked the basic concept of the system but were not satisfied by the quality of the pages in the current prototype. They said that they would like to use it if the system can present more good-looking pages. One participant had little interest in the proposed system, but he is the one who do not care for good design anyway.

6. Discussion

We obtained overall positive feedback from the participants during the user study. They agreed that it is helpful to see existing web pages as a guide and to have a search system that can quickly find pages with desired layouts. However, while the basic concept seems to be well received, the result of the evaluation user study suggests that there still exists room for improvement.

The most frequent complaint from the users is that they sometimes failed to find appropriate web pages they would like to refer. This is because of the limited quality of the database and the matching algorithm.

Improving the Database

We need to improve the size and quality of web pages in the database. The number of web pages used in the user study is 925. This might be too small to cover sufficient variety of layout designs. It is our future work to collect more pages by extending the crawler. It is also important to develop faster matching algorithms to handle large databases.

The quality of the web pages in the database is very important. Users usually ignored "bad-looking" web pages even though those pages matched the user query. The typical tendency was that the higher user's skill is, the more sophisticated designs are required. It is difficult to automatically determine the visual quality of web pages. One way is to use a ranking method such as those used in Google [Goo]. The expectation is that the popular pages

should also have good-looking layout designs. It could also be useful to collect web pages that have higher accessibility. Manual collection of high quality pages is also a viable option because our system does not need massive amount of pages. In this case, the database works as a sort of design templates.

One interesting approach is to improve the quality of the database by natural selection. We can expect that the example pages that are used by many users during the design process have high-quality design. The system can gradually improve the database by constantly evaluating the popularity of the example pages and replacing least popular pages with new web pages retrieved from the WWW.

Improving the Matching Algorithm

The evaluation study in Section 5.3 revealed that our algorithm works well for finding specific pages that the user has in mind but sometimes fails to find pages with similar impressions. This is partly because we currently focus on the positions and size of individual objects on the screen. According to an analysis for good web page design [IHS01], the occupancy ratio and the total number of texts and images in a page has strong impact on the quality of web pages. We plan to incorporate these factors into the matching algorithm. Other possible extensions are to use relative spatial relation among objects using ARG matching [Pet02]. However, it requires exponential time and space. We are interested in testing ImageMap method [HS00][PFL02] that maps all data in a multi-dimensional space in the preprocessing to accelerate the search.

Our current algorithm only takes the position of objects into account and ignores topology. Fonseca introduced a fast search system for vector graphics using topology and geometry matching [Fon04]. They also introduced a multidimensional indexing structure based on eigenvalues. We also would like to test their methods in our system.

Relation to Other Approaches

The user study showed it is useful to see existing web pages as a guide for layout design. Here we would like to discuss other approaches to that end. One is design templates and the other is traditional text search engines.

Design templates are usually provided by a small number of people, which means that the variety of the designs is also limited. The proposed system could provide much more variety of designs. In addition, we believe that the user can learn a lot by observing actual web pages on the net. The advantage of the design templates is that their quality is guaranteed. Our system can also be used as an index for large design templates.

Traditional text search is also useful for finding example web pages. For example, when one wants to design a diary page, one may search the web with the keyword of "diary". We believe that our system can complement these text search engines. As a user in the study mentioned, the process of making web page typically proceeds from concept, then contents and layout. Therefore, one would first search the web using keywords to determine the overall concept and organization, then proceed to search the web using our system to determine the layout.

7. Future Work

In addition to improving the database and the algorithm, we are also exploring several other possibilities to support the web page design process as a whole. We are currently developing a tool that can automatically arrange text and images according to an example layout found in the search. Internally, the system analyzes the logical structure of the example and original pages [BVC04]] and replaces the text and images in the example page with the user's original texts and images. With this semi-automatic conversion, one can explore many design possibilities with her/his own texts and images, which can further improve the efficiency and quality of the web page design.

8. Conclusion

We introduced a system that helps inexperienced users in the design of web page layouts. Using our sketch-based query interface, users can quickly specify the overall design of the desired layout and find corresponding web pages with HTML coding. This helps inexperienced users to learn the capabilities and limitations of the HTML-based layout, and arrive at better designs through iterative exploration with the system.

We focused on web page layouts in this paper. However, the idea of using abundant materials on the net as examples for better design can be applicable to other design tasks such as presentation slides and PDF documents. We hope that this paper encourages similar explorations in other domains.

9. Acknowledgements

We thank test users participated in the study for their help and the members of the User Interface Research group for valuable discussions.

This research is partially supported by IPA mito-youth project with a project manager Prof. Ikuo Takeuchi and project management group Little Studio Inc.

References

[Ask] AskJeeves: http://www.ask.com/

[BVC04] BOUILLON L., VANDERDONCKT J., CHOW K.C.: Flexible Re-engineering of Web Sites, In *Proc. of the international conference on Intelligent User Interfaces*, (2004), 132-139.

[CGFJ02] CAETANO A., GOULART N., FONSECA M., JORGE A.: JavaSketchIt: Issues in Sketching the Look of User Interfaces. In *Proc. of the 2002 AAAI Spring Symposium - Sketch Understanding* (2002), 9-14.

[CHH97] CULLEN J.F., HULL J.J., HART P.E.: Document Image Database Retrieval and Browsing using Texture Analysis. *ICDAR* , (1997) 718-721.

[CSY87] CHANG S. K., SHI Q.Y., YAN C.W.: Iconic Indexing by 2-D Strings. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 9, 3(1987), 413-428.

[FJ01] FONSECA M.J., JORGE J.A.: Experimental Evaluation of an On-line Scribble Recognizer. *Pattern Recognition Letters* 22, 12 (2001), 1311-1319.

[Fli95] FLICKNER M. et. al.: Query By Image and Video Content: The QBIC System. *IEEE Computer*, 28,9 (1995), 23-32.

[FLM98] FLORESCU D., LEVY A., MENDELZON A.: Database Techniques for the World-Wide Web: A Survey. *SIGMOD Record*, 27, 3(1998), 59-74.

[Fon04] FONSECA M.J., *Sketch-Based Retrieval in Large Sets of Drawings*, Ph.D thesis, Univ. Tecnica de Lisboa. (2004). http://immi.inesc-id.pt/~mjf/

[GL85] GOULD J.D., LEWIS C.: Designing for Usability: Key Principles and What Designers Think. *Communications of the ACM*, 28, 3(1985), 300-311.

[Goo] Google: http://www.google.com/

[HC01] HU W.C., CHEN Y.: An Overview of World Wide Web Search Technologies. In *Proc. of 5th World Multi-Conference on System, Cybernetics and Informatics*, (2001).

[HGLS98] HEARST M.A., GROSS M.D., LANDAY J.A., STAHOVICH T.E.: Sketching Intelligent Systems. *IEEE Intelligent Systems*, 13, 3(1998), 10-19.

[HS00] HJALTASON G., SAMET H.: Contractive Embedding Methods for Similarity Searching in Metric Spaces. *Technical Report TR-4102, Computer Science Department*, Univ. of Malyland, (2000).

[IHS01] IVORY M., HEARST M., SINHA R.: Empirically Validated Web Page Design Metrics, *ACM SIGCHI'01 Conference: Human Factors in Computing Systems*, (2001) 53-60.

[LH90] LEE S.Y., HSU F.J.: 2D C-String: A New Spatial Knowledge Representation for Image Database Systems. *Pattern Recognition*, 23, 10(1990), 1077-1087.

[LNHL00] LIN J., NEWMAN M. W., HONG J. I., LANDAY J.A.: DENIM: Finding a Tighter Fit Between Tools and Practice for Web Site Design. In *CHI Letters: Human Factors in Computing Systems*, 2, 1(2000), 510-517.

[MCL97] MARSICOI M. D., CINQUE L., LEVIALDI S.: Indexing pictorial documents by their content: a survey of current techniques. *Image and Vision Computing*. 15, 2 (1997), 119-141.

[MHKF03] MIN P., HALDERMAN J.A., KAZHDAN M., FUNKHOUSER T.A.: Early Experiences with a 3D Model Search Engine. In *Proc. of Web3D Symposium* (2003).

[Pet02] PETRAKIS E.G.M.: Design and Evaluation of Spatial Similarity Approaches for Image Retrieval. *Image and Vision Computing*, 1, 20 (2002), 59-76.

[PFJ00] PINTO-ALBUQUERQUE M., FONSECA M.J., JORGE J.A.: Visual Languages for Sketching Documents. In *Proc. of the IEEE Symposium on Visual Languages* (2000), 225-232.

[PFL02] PETRAKIS E.G.M., FALOUTSOS C., LIN K.L.: ImageMap: An Image Indexing Method Based on Spatial Similarity. *IEEE Transactions on Knowledge and Data Engineering*, 14, 5(2002).

[PO96] PETRAKIS E.G.M., ORPHANOUDAKIS S.C.A.: Generalized Approach to Image Indexing and Retrieval Based on 2-D Strings. *Intelligent Image Database Systems,* World Scientific, (1996), 197-218.

[PSM03] PANT G., SRINIVASAN P., MENCZER F.: Crawling the Web. In M. Levene and A. Poulovassilis, editors: *Web Dynamics*, Springer-Verlag (2004).

[RHC99] RUI Y., HUANG T.S., CHANG S.F.: Image retrieval: current techniques, promising directions and open issues. *Journal of Visual Communication and Image Representation*, 10 (1999), 39-62.

[SM99] SCIASCIO E.D., MONGIELLO M.: Query by Sketch and Relevance Feedback for Content-Based Image Retrieval over the Web. *Journal of Visual Languages and Computing,*10, 6(1999).

[Sch] SCHROEDER W.: Testing Web Sites with Eye-Tracking. *User Interface Engineering*. http://world.std.com/~uieweb/eyetrack1.htm

[SC96] SIMITH J. R., CHANG S. F.: VisualSEEK: A Fully Automated Content Based Image Query System. In *Proc. of ACM Multimedia Conference*, (1996).