Robust Web Page Segmentation for Mobile Terminal Using Content-Distances and Page Layout Information

Gen Hattori
KDDI R&D Laboratories
2-1-15 Ohara Fujimino
Saitama Japan
+81 49 278 7334
gen@kddilabs.jp

Keiichiro Hoashi
KDDI R&D Laboratories
2-1-15 Ohara Fujimino
Saitama Japan
+81 49 278 7332
hoashi@kddilabs.jp

Kazunori Matsumoto
KDDI R&D Laboratories
2-1-15 Ohara Fujimino
Saitama Japan
+81 49 278 7329
matsu@kddilabs.jp

Fumiaki Sugaya
KDDI R&D Laboratories
2-1-15 Ohara Fujimino
Saitama Japan
+81 49 278 7360
fsugaya@kddilabs.jp

ABSTRACT

The demand of browsing information from general Web pages using a mobile phone is increasing. However, since the majority of Web pages on the Internet are optimized for browsing from PCs, it is difficult for mobile phone users to obtain sufficient information from the Web. Therefore, a method to reconstruct PC-optimized Web pages for mobile phone users is essential. An example approach is to segment the Web page based on its structure, and utilize the hierarchy of the content element to regenerate a page suitable for mobile phone browsing. In our previous work, we have examined a robust automatic Web page segmentation scheme which uses the distance between content elements based on the relative HTML tag hierarchy, i.e., the number and depth of HTML tags in Web pages. However, this scheme has a problem that the content-distance based on the order of HTML tags does not always correspond to the intuitional distance between content elements on the actual layout of a Web page.

In this paper, we propose a hybrid segmentation method which segments Web pages based on both the content-distance calculated by the previous scheme, and a novel approach which utilizes Web page layout information. Experiments conducted to evaluate the accuracy of Web page segmentation results prove that the proposed method can segment Web pages more accurately than conventional methods. Furthermore, implementation and evaluation of our system on the mobile phone prove that our method can realize superior usability compared to commercial Web browsers.

Categories and Subject Descriptors
H.4.3 [**Communications Applications**]: *Information browsers.*

General Terms
Algorithms, Performance, Experimentation.

Keywords: Web Page Segmentation, Mobile Phone, Content-Distance, Web Page Layout

1. INTRODUCTION

The number of mobile phone contractors in the world is predicted to exceed 2.5 billion by 2009 [23]. 3G terminals are spreading rapidly in countries such as Japan and Korea, and most of them equip functions which utilize the various features of 3G terminals. One representative feature of such terminals is the mobile phone Web browser. Users are able to download various content elements such as text information, music, pictures and movies from the Web, which can all be played or displayed on their mobile phones via the Web browser.

However, while it is possible for mobile phone users to obtain various information on their terminals, the amount of information on Web sites specialized for mobile phones are restricted, mainly due to the limitations of the user interface of mobile phones. Therefore, there is an increasing demand to efficiently browse general Web pages designed for PCs on the mobile phone. When such demands are met, it will become possible for mobile phone users to obtain the same amount of information as on their PCs. However, since general Web pages are designed to be browsed on a PC with a rich user interface, it is difficult for such pages to be displayed on the small display of mobile phones. Therefore, various methods to convert the PC site to fit the mobile phone have been developed in recent years, and mobile phone Web browsers with such functions, such as the Opera browser, have been commercialized[4]. The usability of such browsers, though, is still poor, since they require a laborious scrolling operation, and are not capable of displaying all Web pages efficiently on the mobile phone.

In previous work, we have proposed a Web page segmentation method, which divides a Web page into two or more small objects in order to make it viewable on a mobile phone[1]. A concept of this method is shown in Figure 1. This method calculates "content-distances," a measure which expresses the strength of connections between content elements of the Web page based on the structural depth of HTML tags (details are described in 3.1), and segments Web pages at the positions where the calculated content-distance exceeds a predefined threshold. Since this method does not explicitly utilize HTML grammar for analysis of the Web page, it is robust to frequently observed HTML grammar errors, thus, is capable of segmenting any Web page. Furthermore, we have also developed a method to dynamically calculate an optimal threshold to segment the Web page, based on the distribution of content-distance. However, further analysis have made clear that, while the average accuracy of Web page segmentation has been improved by these methods, as written in References [2], the content-distance calculated based on the order of HTML and the intuitional distance between content elements of the Web page do not necessarily correspond, which is a cause of unsatisfactory segmentation results.

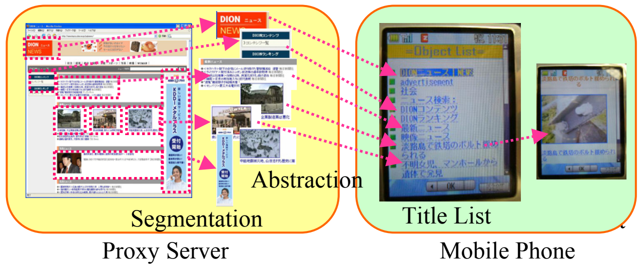


Figure 1. A Concept of the Segmentation for Mobile Phone

In this paper, we propose a hybrid Web page segmentation method, which combines the previously proposed method based on content-distance, with a novel function which conducts preliminary segmentation based on the analysis of the Web page layout. The proposed method is expected to significantly improve the accuracy of Web page segmentation, and the usability of mobile phone Web browsers, when its segmentation results are applied.

The structure of this paper is as follows. Firstly, in Chapter 2, we describe related works on Web browsing by the mobile phone, and clarify the requirements based on analysis of existing problems. In Chapter 3, we present a brief description of our previously proposed Web page segmentation method based on content-distance. In Chapter 4, we propose the hybrid Web page segmentation method. In Chapter 5, we describe the results of the evaluation experiment, introduce an implementation example of Web page segmentation system for mobile phone, and present results of a usability evaluation. Finally, we provide our conclusions in Chapter 6.

2. RELATED WORKS

In this section, we describe conventional efforts to enable efficient browsing of PC-based Web pages on the mobile phone, and clarify the requirements of such methods based on analysis of the problems of conventional methods.

Related work in this area can be classified into three techniques. One is to modify the layout of the Web page so that the entire content can be displayed on the mobile phone as much as possible. The second is to “restructure” the Web page to suit the mobile phone. In this case, reconstruction of the Web page includes the omission of contents included in the original Web page, if necessary. The last is to zoom the thumbnail image of a Web page in order to read details of specified part. Explanations of related works in each technique, as well as their known problems are presented as follows.

(1) Web page layout modification

One method to change the layout of a Web page is to adjust the width of the page to fit the small display of the mobile phone. This method is implemented in existing mobile phone Web browsers such as Opera[4]. In this case, the Web page layout is transformed into a vertically long form. An advantage of this method is that it is capable of generating an appearance similar to the original appearance on PCs. However, in a situation where information of the users’ interest is located at the bottom area of the transformed Web page, the user needs to repeatedly scroll down on their mobile phone, which is obviously inconvenient. This is assumed to be a serious problem, since typical users often are only interested in a specific part of the Web page [3].

Moreover, there have been methods proposed which are specialized to transform table-formatted information for the mobile phone [5][6]. Such methods automatically extract item names from the table, and display each column along with the extracted item names, in order to create an easy-to-view display for the user. However, these methods cannot be applied to transform information which is not written in the table form.

In addition to the above methods, there are methods to utilize analysis results of the “site map” of the Web site to automatically generate a menu-like page for the mobile phone [7][8], and methods which generate a list of contents based on a user-defined parameter which expresses the significance of hyperlinks in the Web page[9][10]. However, the former method obviously will not work for Web pages with no site maps. The latter method is also difficult for mobile phone usage, since it requires the user to view the contents of the linked Web pages to define the significance parameter.

(2) Web page restructuring

There have been a number of methods proposed to restructure Web pages for mobile phone browsing. One approach is to create a set of “objects,” i.e., a set of highly correlated content elements (such as an image and its description) based on the analysis of the layered structure of HTML tags[11][12][13][14][15]. The set of objects can be utilized to enable browsing of Web pages with abundant information on a small display, for example, by presenting each object once at a time. However, since these methods conduct Web page segmentation by analyzing the structure of HTML, the page is required to be grammatically correct. Naturally, this method is not capable of achieving accurate segmentation of Web pages with HTML grammar errors, which are frequently observed on the Web.

Moreover, there is a method to generate a digest of the Web page by deletion of portions of the Web page that do not suit the user's preference, which is estimated by counting the appearance frequency of words in the user's access log[16]. While this method is capable to automatically shrink the content of the Web page, it does not assure that the resulting HTML is small enough for browsing on the mobile phone.

Furthermore, a Web page segmentation method has been proposed which allocates objects on the screen to two or more mobile phones to enable browsing the entire content of the page by cooperation of others’ mobile phones[17]. However, this method requires the HTML structure of the Web page to be converted to a complete graph with weight parameters; hence, is also prone to Web pages containing HTML grammatical errors.

(3) Web page zooming

There are also methods which utilize a thumbnail image of the Web page. In this approach, users select specific parts of the Web page, which are then magnified by the system [18][19][20]. While this approach has been proved to be effective for PDAs, which have a relatively large display and stylus, its implementation on the mobile phone is problematic, due to the interface limitations, namely, the smaller screen and button-based operation.

3. SYSTEM REQUIREMENTS

Based on the analysis of the problems of the previously described conventional methods, we have designed the structure of the Web viewer for mobile phones, which is shown in Figure 2. The proposed system is composed of the following four functions.

+ Request Reception: a function to accept user requests
+ Browser Emulator: a function to acquire HTML of Web site requested by the user
+ Segmentation Processor: a function which automatically divides acquired HTML
+ XHTML Rebuilder: a function to generate new XHTML based on segmentation results

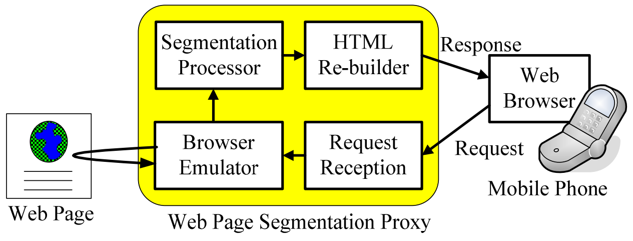


Figure 2. Web Viewer for Mobile Phone

In this paper, we focus on the methodology to realize the Segmentation Processor. Based on the previously described problems of conventional methods, we summarize the system requirements for a practical Web viewer for mobile phones to the following two items:

Requirement 1: Automatic segmentation of Web page information

> Due to the limitations of mobile phone user interfaces, an effective system should be able to automatically limit the volume of information to be displayed.

Requirement 2: Capability to correspond to all HTML

> An effective system should be able to segment all Web pages, including pages with grammatical HTML errors.

4. WEB PAGE SEGMENTATION USING CONTENT-DISTANCE

In this section, we describe our previously proposed Web page segmentation method based on content-distance [1]. This method calculates content-distance by analyzing HTML based on a signal processing-like approach, and divides the Web page into two or more objects. The segmentation results can be utilized to restructure Web pages for the mobile phone, by selecting objects which should be displayed. This method satisfies the two previously mentioned requirements set in the previous section.

4.1 Segmentation Algorithm Based on Content-Distances

The main feature of our previously proposed method is that it utilizes only the number of tags and their relative depth to segment the Web page. In other words, this method does not depend on HTML grammar or tag types used in the Web page. Therefore, the robustness of this method to Web pages written in irregular HTML is higher than that of other existing methods, such as methods of Chen [11] and Maekawa et al.[12], which are dependent on the assumption that the tree structure of the entire Web page is grammatically correct.

In Reference [1], we examined the ratio of HTML with fatal errors that cannot be restored automatically, by using Tidy [21], a restoration tool of irregular HTML, *i.e.*, Web pages which contain HTML grammatical errors. Examination results showed that the ratio of Web pages with irregular HTML was 8.5%-27.1% which varies among site categories, *e.g.* news, entertainment, and sports. Although some HTML parsers may be capable of disregarding erroneous HTML tags, this high ratio of irregular HTML occurrence indicates the necessity of a robust Web segmentation method.

In the proposed method, we define the "content element" of a Web page as one of the following four types of data (a)-(d). CSS (Cascade Style Sheet) are not explicitly taken up in this method, since the effect of style information to our algorithm is minimal.

(a) Anchor specified with <A> tag
(b) Image specified with <IMG> tag
(c) JavaScript with <SCRIPT> tag
(d) All text data other than (a)-(c)

The proposed Web page segmentation method based on content-distance consists of the following three steps: (A) Content element extraction, (B) Content-distance calculation, and (C) Web page segmentation. Details of each step are explained as follows:

(A) Content element extraction

First, the system extracts content elements from the HTML source of the Web page. In addition, the system deletes tags which are included for modification and arrangement of content elements, such as font tag (<FONT>) and layout tag (<TABLE>), since the expressions of such tags are often difficult to be displayed on mobile phones.

(B) Content-distance calculation

The system calculates the "content-distance," i.e., the distance between content elements, based on the number and depth of tags that are inserted between adjacent content elements. Tag depths are calculated as follows. First, the initial value of all tags is set to 0. Next, if the tag is an opening tag, we add 1 to the depth of the tag preceding it; if the tag is a closing tag, we subtract 1 from the depth of the preceding tag. The concepts of tag depth and content-distance are shown in Figure 3. The horizontal axis x in the lower graph expresses the tag sequence order, which starts from the <HTML> tag at the head of the HTML source. The vertical axis y expresses the tag depth.

Content-distance between content elements **a** and **b** is defined as $S_{(\mathbf{a},\mathbf{b})}$, which is calculated by Formula 1:

$$S_{(a,b)} = \max\left\{\sum_{i=x_a}^{x_b}\left|\max\{y_a, y_b\} - f_{ab}(i)\right|, \sum_{i=x_a}^{x_b}\left|\min\{y_a, y_b\} - f_{ab}(i)\right|\right\} \quad (1)$$

where x_a and x_b denote the sequence orders of the tags of content elements **a** and content **b,** respectively. Similarly, y_a and y_b denote the depth of the tags of content elements **a** and **b,** respectively. Moreover, $f_{ab}(i)$ indicates the value of y between content elements **a** and **b** where $x=i$. Note that variables x_a, x_b, y_a, y_b, $f_{ab}(i)$ in Formula 1 correspond to Figure 3.

(C) Web page segmentation

The system divides the Web page into two or more objects, by comparing the calculated content-distance with a predefined threshold. This procedure is recursively repeated. This procedure is described as following steps. Let $S_{\max}$ and S_{average} denote the maximum value and average of content-distance in an object, respectively. Note that parameters N_1 and N_2 always meet the condition $N_1 > N_2$.

(Step 1) In the initial state, the entire Web page is considered as one object (*ObjectID*=root).

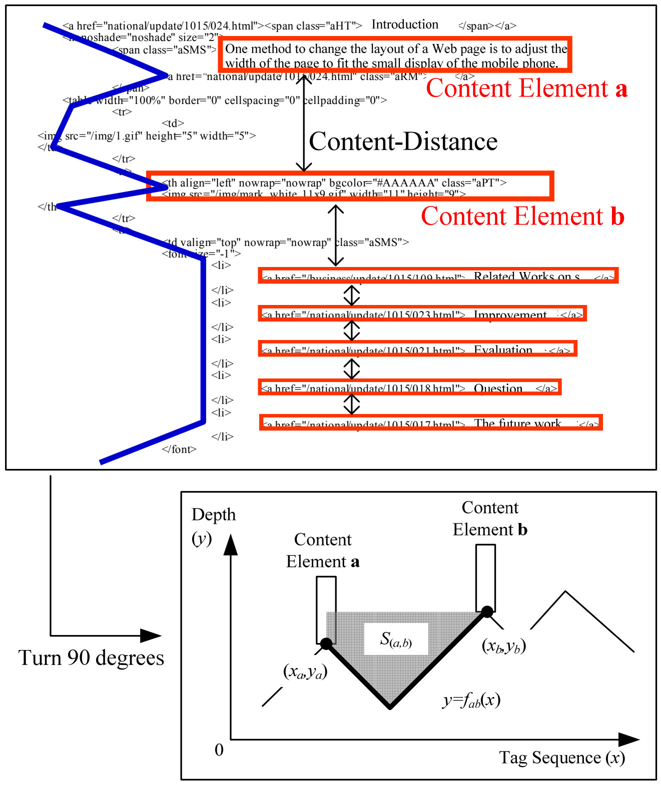
(Step 2) If $S_{\max} > N_1 * S_{\text{average}}$, the system divides this object at the position of $S_{\max}$.

(Step 3) Else, if the minimum number of tags in a group when it is segmented exceeds M, and $S_{\max} > N_2 * S_{\text{average}}$, the system divides this object at the position of $S_{\max}$.

(Step 4) If the object is divided in (Step 2) or (Step 3), *ObjectID* moves to the left object of its division result and the process returns to (Step 2). Otherwise, the process advances to (Step 5).

(Step 5) When the object processed in (Step 2) or (Step 3) is a left object, *ObjectID* moves to the right object and the process returns to (Step 2). Otherwise, the process advances to (Step 6).

(Step 6) When the object processed in (Step 2) or (Step 3) is a right object and *ObjectID* != root, *ObjectID* moves to the parent object and the process returns to (Step 5). Otherwise, the process is the end.



```
<a href="national/update/1015/024.html"><span class="aHT">  Introduction     </span></a>
<hr noshade="noshade" size="2">
   <span class="aSMS">  One method to change the layout of a Web page is to adjust the
                        width of the page to fit the small display of the mobile phone.
          <a href="national/update/1015/024.html" class="aRM">     </a>
   </span>
<table width="100%" border="0" cellspacing="0" cellpadding="0">
   <tr>
         <td>
<img src="/img/1.gif" height="5" width="5">
</td>
   </tr>
   <tr>
         <th align="left" nowrap="nowrap" bgcolor="#AAAAAA" class="aPT">
         <img src="/img/mark_white_11x9.gif" width="11" height="9">
</th>
   </tr>
   <tr>
         <td valign="top" nowrap="nowrap" class="aSMS">
         <font size="-1">
              <li>
                    <a href="/business/update/1015/109.html">  Related Works on s   </a>
              </li>
              <li>
                    <a href="/national/update/1015/023.html">  Improvement   </a>
              </li>
              <li>
                    <a href="/national/update/1015/021.html">  Evaluation   </a>
              </li>
              <li>
                    <a href="/national/update/1015/018.html">  Question   </a>
              </li>
              <li>
                    <a href="/national/update/1015/017.html">  The future work   </a>
              </li>
         </font>
```

Figure 3. Tag Depth and Content-Distances

4.2 Segmentation-Threshold Estimation with Standard Deviation of Content-Distances

The optimal values of N_1 and N_2 in the previous method are obviously different for each Web page. However, it is difficult to manually define optimal values for all Web pages. In order to solve this problem, we have developed an algorithm which calculates the optimal value of N_1 and N_2 for each Web page dynamically, based on the standard deviation of content-distance. Details of this method are as follows.

(A) Definition of baseline values

(Step A1) Select an arbitrary Web page to be used as a standard. Let N_{t1} and N_{t2} note the optimal values of N_1 and N_2 for the targeted Web page, respectively.

(Step A2) Let N_{b1} and N_{b2} note empirically selected values of N_1 and N_2 for the standard page, respectively.

(Step A3) Derive sets of content-distances $S_{b(i,i+1)}$ on standard page according to procedure (C) of 3.1.

(Step A4) Calculate σ_{Sb} according to Formula 2.

$$\sigma_{Sb} = \sqrt{\frac{\sum_{i=1}^{n_b-1}(\bar{S}_b - S_{b(i,i+1)})^2}{n_b - 1}} \qquad (2)$$

where:

$\bar{S}_b$	: Mean value of the content-distance of based page
$S_{b(i,i+1)}$	: Content-distance between content element i and content element i+1 on based page
n_b	: Number of content elements extracted on based page

(B) Calculation of N_{t1} and N_{t2} for the target Web page

(Step B1) Derive sets of content-distances $S_{t(i,i+1)}$ on a target page according to procedure (C) of 3.1.

(Step B2) Calculate σ_{St} according to Formula 3.

(Step B3) Calculate N_{t1} and N_{t2} according to Formulae 4 and 5. α is a positive real number α>0.

$$\sigma_{St} = \sqrt{\frac{\sum_{i=1}^{n_t-1}(\bar{S}_t - S_{t(i,i+1)})^2}{n_t - 1}} \qquad (3)$$

$$N_{t1} = N_{b1} + N_{b1} * \left(\frac{\sigma_{St}}{\sigma_{Sb}} - 1\right) * \alpha \qquad (4)$$

$$N_{t2} = N_{b2} + N_{b2} * \left(\frac{\sigma_{St}}{\sigma_{Sb}} - 1\right) * \alpha \qquad (5)$$

where:

$\bar{S}_t$	: Mean value of the content-distance of targeted page
$S_{t(i,i+1)}$	: Content-distance between content i and content i+1 on targeted page
n_t	: Number of contents extracted on targeted page

4.3 Problems

Since the above segmentation methods are capable of segmenting any Web page, regardless of the existence of HTML grammar errors, it is clear that these methods satisfy the two system requirements presented in Section 2.2. However, the following two new problems become apparent.

(Problem 1)

Content-distance calculated based on the order of HTML and the intuitional distance between content elements of the Web page do not necessarily correspond. For instance, in the case of <TABLE> tags, the cell at the right edge of the first column and the cell at the left edge on the second column are adjacent in the source of HTML; however, these two cells are actually located far away from each other when the page is displayed on a Web browser.

(Problem 2)

Most news sites and portal sites adopt layouts which consist of separate components, *e.g.*, a header, footer, and side menu, etc. in order to improve readability. Naturally, the tag structures of each component are different from each other. However, the previous method cannot correspond to those partial features, since it defines a single optimal value for N_1 and N_2 for each Web page.

5. HYBRID SEGMENTATION

To solve the problems described above, we propose a hybrid Web page segmentation method using both content-distance and page layout information. The proposed method solves the two problems in two steps. Firstly, in order to obtain abstract information of the Web page layout, the system executes a preliminary segmentation process by searching tags which are related to the page layout. Next, the system individually sets the value of N_1 and N_2 for each roughly extracted segment of HTML (hereafter referred to as a "sub-object") generated by the preliminary segmentation process.

The implementation of preliminary segmentation based on Web page layout analysis is expected to solve the problems of the previously proposed method, since it explicitly utilizes the layout of the Web page, thus is capable to extract sub-objects according to their actual location when displayed on a PC Web browser. Furthermore, since the proposed method defines an optimal threshold for each extracted sub-object, it is capable to conduct accurate segmentation regardless of differences between the tag structures of separate sub-objects.

However, since the preliminary segmentation process requires the HTML which expresses the Web page layout to be grammatically correct, this process is affected by HTML grammar errors. Therefore, the sub-object segmentation process is applied to the Web page only when the layout information is analyzed correctly. In cases where errors occur in the layout analysis process, the hybrid system applies only the previous method based on content-distance.

5.1 Hybrid Segmentation System Architecture

The process flow and system architecture of the hybrid segmentation system is shown in Figure 4. As clear from this Figure, this system is composed of four components: the Layout Analyzer, the Layout based Segmentation Engine (LSE), Content-distance based Segmentation Engine (CSE), and XHTML Re-builder. The Layout Analyzer detects tags which define page layout. LSE conducts preliminary segmentation based on the output of the Layout Analyzer. CSE conducts segmentation based on content-distance, both for the sub-objects extracted by LSE, and for HTML which do not contain layout defining tags. XHTML Re-builder builds the final XHTML.

Tags which are frequently used to define page layout are the <TABLE>, <DIV> and <FRAME> tags. It is easy to extract the structure of the Web page constructed by <FRAME> tags, since independent HTML files are allocated within each sub-frame. Therefore, we will focus on the description of handling Web pages which contain <TABLE> or <DIV> tags. Detailed explanation of the hybrid segmentation procedure is as follows.

(Step 1) Layout Analyzer acquires HTML of the target site, and searches the presence of <TABLE> or <DIV> tags.

(Step 2) If Layout Analyzer finds <TABLE> or <DIV> tag in the HTML, it sends the HTML to Layout based Segmentation Engine (LSE), and proceeds to (Step 3). Else, it sends the HTML to Content based Segmentation Engine (CSE), proceeds to (Step 4).

(Step 3) LSE divides the HTML using <TABLE> or <DIV> tags, sends each sub-object to CSE, and proceeds to Step 4. Details of the segmentation method of LSE will be explained in Section 4.2. Once LSE has received all results from CSE, it sends them to XHTML Re-builder.

(Step 4) CSE conducts segmentation of the received HTML or object, based on content-distance based method, and returns the results to LSE (and proceeds to (Step 3)) or sends them to XHTML Re-builder. (proceeds to (Step 5))

(Step 5) XHTML Re-builder rebuilds the received segmentation results for mobile phones and outputs the final XHTML.

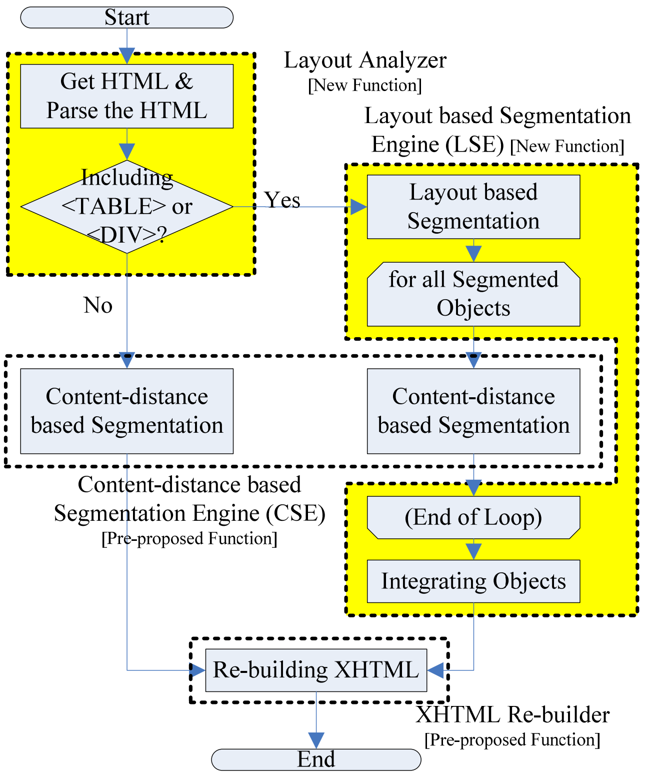

Start
Layout Analyzer
[New Function]
Get HTML & Parse the HTML
Including <TABLE> or <DIV>?
Yes
No
Layout based Segmentation Engine (LSE) [New Function]
Layout based Segmentation
for all Segmented Objects
Content-distance based Segmentation
Content-distance based Segmentation
Content-distance based Segmentation Engine (CSE)
[Pre-proposed Function]
(End of Loop)
Integrating Objects
Re-building XHTML
XHTML Re-builder
[Pre-proposed Function]
End

Figure 4. Process Flow

5.2 Layout-based Segmentation Algorithm

The LSE analyzes the <TABLE>, <TR>, <TD>, and <DIV> tags, which are not only used to compose the table, but also to define the layout of the Web page. Namely, LSE measures the size of each cell in the table based on pixels, and judges whether to divide the cell by comparing the calculated size with the maximum cell size $P_{\max}$. The division result is sent to CSE. The process flow of LSE is shown in Figure 5, and the detailed processing steps are described as follows.

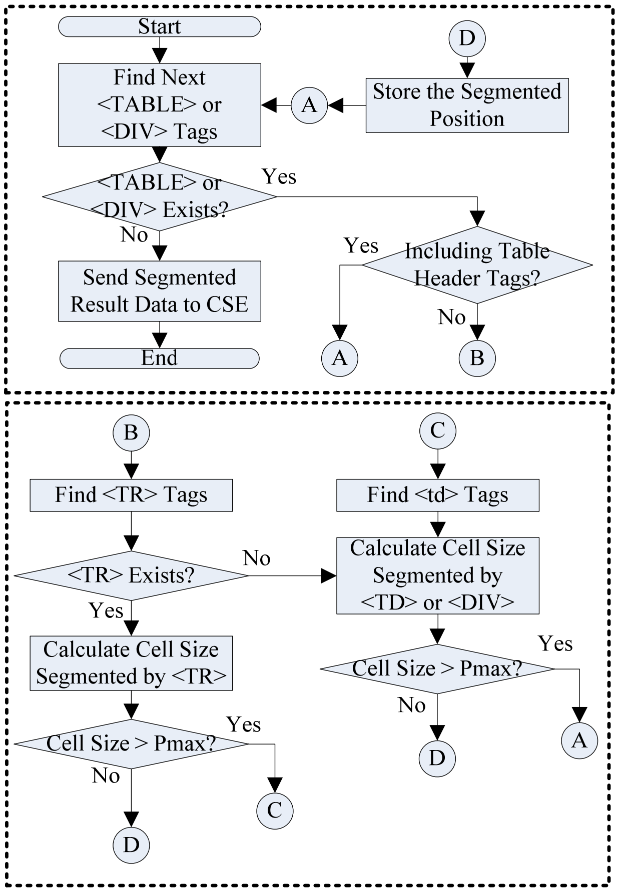
(Step 1) LSE searches for the first <TABLE> or <DIV> tag from the pointer position on the HTML source (the initial pointer position is the head of the HTML). If LSE cannot find a <TABLE> or <DIV> tag, LSE sends the current division result to CSE. If no <TABLE> or <DIV> tag exists, LSE sends the message "no segmentation" to CSE.

(Step 2) When LSE detects a <TABLE> tag, and also finds that the <TABLE> tag includes child tags related to table formation such as <THEAD>, <TFOOT>, and <TH>, LSE judges that the <TABLE> tag is not used for layout definition, and returns to (Step 1) to retrieve the next <TABLE> tag. When LSE detects a <DIV> tag, the process advances to (Step 4).

(Step 3) If the <TABLE> tag does not contain tags related to the table header, then LSE searches for the <TR> and <TD> tags. If the minimum size of the cell segmented by <TR> and <TD> is smaller than $P_{\max}$, LSE terminates segmentation. The process returns to (Step 1) to retrieve the next table tag after storing the segmentation position.

(Step 4) If the minimum size of the cell segmented by <DIV> is smaller than $P_{\max}$, LSE terminates segmentation. The process returns to (Step 1) to retrieve the next table tag after storing the segmentation position.

The method to calculate the size of the cell is based on the following sub-steps:


Start
Find Next <TABLE> or <DIV> Tags
<TABLE> or <DIV> Exists?
Yes
No
Send Segmented Result Data to CSE
End
D
Store the Segmented Position
A
Including Table Header Tags?
Yes
No
A
B
B
Find <TR> Tags
<TR> Exists?
No
Yes
Calculate Cell Size Segmented by <TR>
Cell Size > Pmax?
Yes
No
C
D
C
Find <td> Tags
Calculate Cell Size Segmented by <TD> or <DIV>
Cell Size > Pmax?
Yes
No
D
A

Pmax: Maximum Cell Size (pixel)

Figure 5. Layout-based Segmentation Algorithm Flow

(Sub-step 1) For tags which have “width” and “height” attributes (e.g., <IMG>, <OBJECT>, <APPLET>, <LAYER>, and <EMBED> etc.), the area of the cell is calculated based on the attribute values (pixels).

(Sub-step 2) For tags which have “size” attributes (e.g., <INPUT> and <SELECT> etc.), the value of the “size” or “maxlength” attributes, and the number of text characters is used to calculate the pixel size of the cell. For the button of the <INPUT> tag, the character string length of the “value” attribute is used for pixel value calculation.

(Sub-step 3) The size of the text is calculated from the number of characters and the font size. The display resolution is assumed to be standard 96dpi. Size differences of various font designs are ignored, since such differences are extremely small.

(Sub-step 4) Empty spaces caused by the difference of the content of cells are also ignored, since the purpose of LSE is extraction of a rough layout. And $P_{\max}$ takes a comparatively large value, for example, as much as 1/5-1/10 of the entire page.

6. EXPERIMENTS

To confirm the effectiveness of the proposed method, we conducted experiments to evaluate the segmentation accuracy, its usability, and the performance on mobile phones. In order to evaluate accuracy, we compared the results of the conventional and proposed methods by conducting segmentation experiments on popular Web sites. Direct comparison of our method to related work described in Section 2.1 is difficult, since the evaluation of related work have been conducted subjectively, e.g., by user questionnaires. Therefore, we evaluate the effectiveness of our proposed method by comparing the segmentation results with our previously proposed method. Furthermore, we evaluated the usability of the Web page restructuring results on the mobile phone, by comparing our implementation with a commercial Web page transcoder [25]. Details of the experimental conditions, accuracy evaluation, and usability experiments are presented in the following sections.

6.1 Accuracy Evaluation Experiments

6.1.1 Experimental conditions

First, in order to evaluate the segmentation accuracy of the proposed method, we compare its results with those of the conventional method. Hereafter, the conventional content-distance based method and the proposed hybrid method are referred to as “*CD*” (*C*ontent-*D*istance) and “*HYB*” (*HYB*rid), respectively.

The 100 Web sites in the United States and Japan are selected for our evaluation, based on information on the Alexa Web site [24], where Web site rankings based on their traffic is opened to the public. Web sites where JavaScript and CSS are frequently used are omitted from the evaluation set. An extract of the Web site used for our experiments is presented in Table 1.

In order to conduct experiments, we implemented the *CD* and *HYB* methods on the PC, acquired HTML from the Web sites in the evaluation set, and conducted segmentation for both methods. The parameter settings of the two methods in our experiments are shown in Table 2.

Table 1. Example of Web Sites Used in Evaluation (The numbers in this Table do not necessarily correspond to the rankings of Alexa site.)

	United States (US)		Japan (JP)
1	Yahoo!	1	Yahoo! Japan
2	Google	2	Google Japan
3	Myspace	3	Mixi
4	MSN	4	Rakuten Ichiba
5	Ebay	5	Livedoor
6	Amazon	6	FC2
7	Youtube	7	Goo Dictionary
8	Craigslist	8	Amazon JP
9	Wikipedia	9	Infoseek Japan
10	CNN	10	@nifty
11	LinkShare	11	MSN Japan
12	Thefacebook	12	Wikipedia
13	TypePad	13	Asahi.com
14	Blogger	14	DION
15	Target	15	Hatena

Table 2. Parameter Settings

N_{b1}	N_{b2}	M	α	$P_{\max}$
3.4	2.3	2	0.36	156000

For evaluation measures, precision and recall of the segmentation results of each method were calculated according to Formulae 6 and 7. "Correct" in Formulae 6 and 7 indicates that the automatically determined segmentation positions by each method are in agreement with the subjectively defined segmentation positions judged by human assessors.

$$\text{Precision} = \frac{\text{(a) Number of "Correct" Segments}}{\text{(b) Number of All Segments}} \quad (6)$$

$$\text{Recall} = \frac{\text{(a) Number of "Correct" Segments}}{\text{(c) Number of All "Correct" Segments}} \quad (7)$$

(a) Number of “Correct” Segments
The number of positions segmented correctly.
(b) Number of All Segments.
The number of all positions segmented in spite of correct or not.
(c) Number of All “Correct” Segments
The number of all positions segmented correctly.

Furthermore, in order to measure the overall accuracy of segmentation, we calculated F-measure based on the precision and recall. The mean F-measure is used to compare our results.

6.1.2 Results

The comparison of the results of two methods is shown in Table 3. Furthermore, significant examples of improved and degraded experiment results are shown in Table 4.

Table 3. Evaluation Results

	Method	Precision	Recall	F-measure
US	***CD***	0.74	0.55	0.63
	HYB	0.77	0.70	**0.73**
JP	***CD***	0.82	0.62	0.71
	HYB	0.82	0.69	**0.75**

Table 4. Significant Results

	Site Name	F-measure (*CD*)	F-measure (*HYB*)
Significant Improvement	Target	0.35	0.96
Degenerated Site	LinkShare	0.77	0.77

The results in Table 3 indicate that, for both JP and US sites, the *HYB* method has achieved higher F-measure compared to the *CD* method, proving the effectiveness of the proposed method. As presented in Table 4, significant improvement was observed for TypePad site and Target site (Item 13 and 15 in US list of Table 1), where the F-measure values improved from 0.57 to 0.86 and from 0.35 to 0.96 respectively. An example of the segmentation result of Target site is shown in Figure 6. In this figure, the dotted rectangle corresponds to the result of Method 1, and the solid rectangle corresponds to that of Method 2. As clear from this Figure, the proposed method was able to correctly divide the Web page into thirteen objects. Moreover, the previously proposed method has loosely divided whole of the page, while the proposed method was able to detect all segmentation points, except for one point (X) located in the left menu part of Figure 6.

On the other hand, examples of erroneous segmentation points on the LinkShare site (Item 11 in US list of Table 1) are illustrated in Figure 7. This example shows that the proposed method could not divide at the point (Y), while the previously proposed method has successfully detected this segmentation point. The cause of this result is assumed to be the parameter setting of $P_{\max}$. Namely, the value of $P_{\max}$ was not optimal for this part of the site, which was actually composed by tables. In this case, this hybrid algorithm cannot set optimal values of N_1 and N_2, since N_1 and N_2 are calculated based on the content-distance within each sub-object obtained from the layout-based segmentation process, and is not capable of defining a suitable value for both side. Furthermore, also in the case of LinkShare, the proposed method could not detect the segmentation point (Z), since the LSE of the proposed method judged the area too small to be divided based on $P_{\max}$. These problems are expected to be solved by development of a method to adaptively decide the optimal value of $P_{\max}$, which we plan to pursue in the near future.

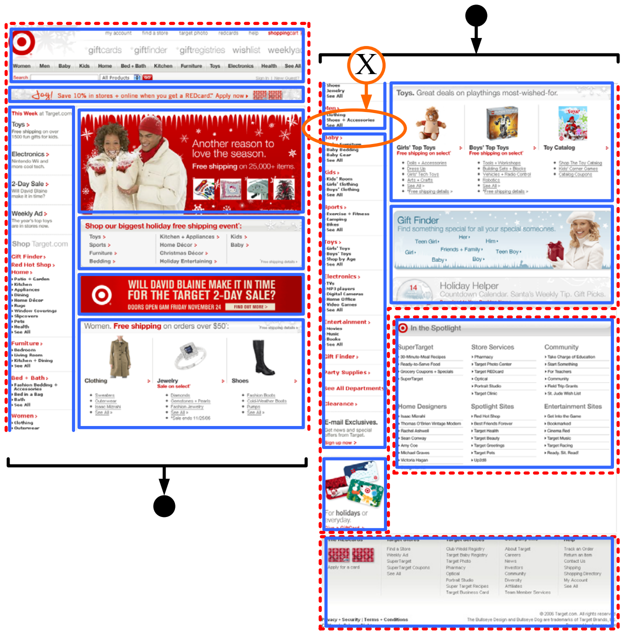


Figure 6. An Example of Advanced Segmentation Result (*CD*: dotted rectangle, *HYB*: solid rectangle)

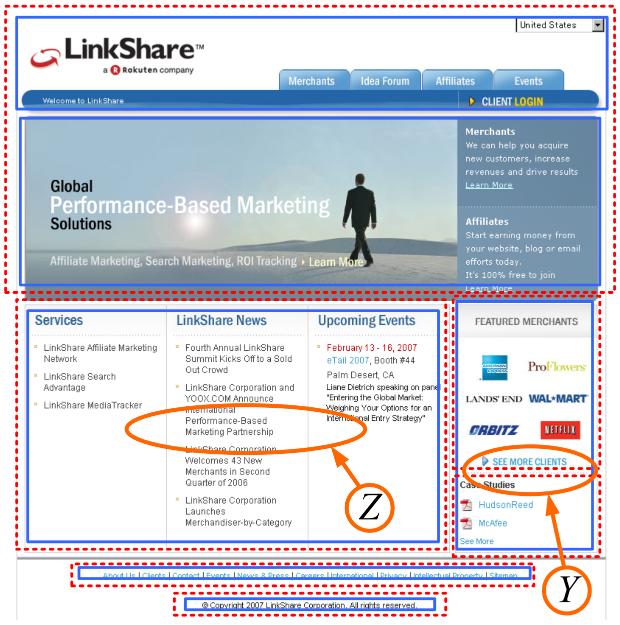


Figure 7. An Example of Erroneous Segmentation Results (*CD*: dotted rectangle, *HYB*: solid rectangle)

6.2 Implementation Example and Usability Evaluation

6.2.1 Implementation

Next, we present an implementation example of our system. Namely, we conducted segmentation of news sites featuring text and photographs, by using the system illustrated in Figure 2. An example of the news site and its segmentation result [22] is shown in Figure 8. Our system was able to correctly recognize the photograph and text shown in the dotted rectangular area, as an object. Figure 9 illustrates how the detected objects are displayed on the mobile phone.

The left figure in Figure 9 shows an example of an object title list, which is generated from the Web page segmentation results of the page illustrated in Figure 8. Titles of each object in this list are generated from the text character strings which appear firstly in each object. The right figure in Figure 9 shows an example of displaying an object, which can be browsed by clicking one of the hyperlinks in the title list. As clear from this Figure, users are able to easily view information of their interest, by minimal scrolling and clicking operations on the automatically generated object title list. Therefore, it is clear that our system provides an extremely efficient interface to the user, especially when compared to conventional Web page layout modification methods explained in Section 2.1, where the user needs to scroll all the way down to the position where information of their interest is located.

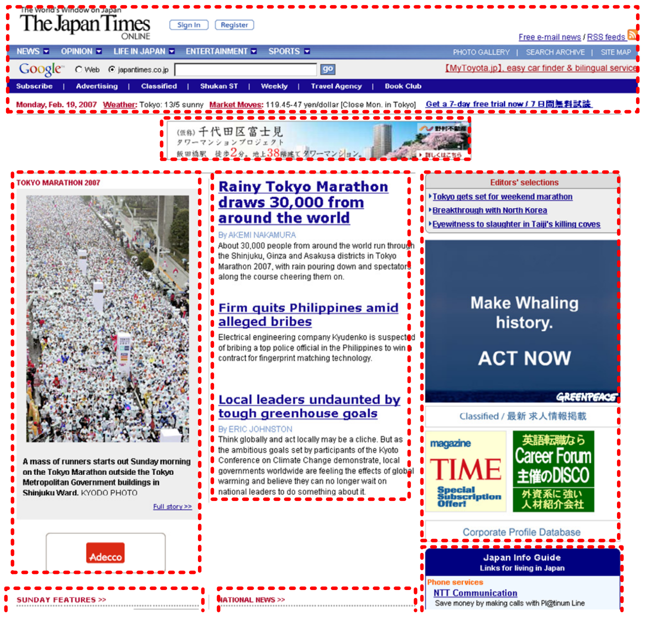


Figure 8. Example of a segmented Web page

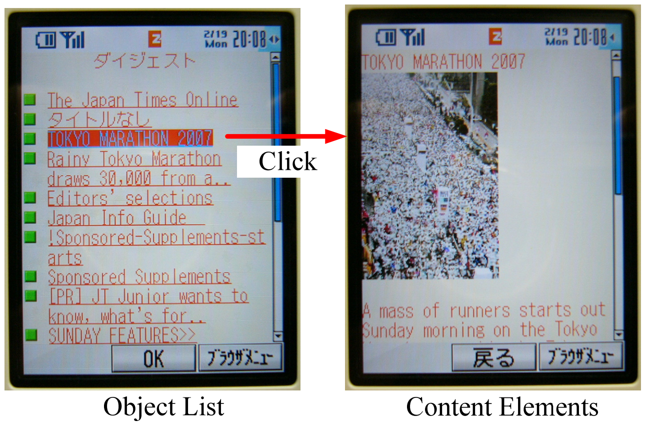


Figure 9. Example of Title List and Content Element Display

6.2.2 Usability Experimental Conditions

Next, to evaluate the usability of our system, we simulated a situation where potential mobile phone users wish to browse specific information located in various positions of the Web page. In order to evaluate usability, we calculated the estimated time necessary to display the required information on their terminals; since time is the most influential factor of the usability for Web browsing on mobile phones. The results are compared with those of the Google Wireless Transcoder [25].

Details of this experiment are as follows. First, we selected five Web sites (Items 1, 4, 5, 13, and 14 of JP in Table 1) which contain abundant information. Next, we set imaginary "target" content elements for each Web site, which represent information of interest of mobile phone users. The target content elements are selected from the top, middle, and bottom parts of each Web page.

The estimated time to display target contents on the mobile phone is calculated based on the predefined time assumed necessary for user operations, namely, the down-scroll and hyperlink tracing operations. For our experiments, we assumed the down-scroll operation time, *i.e.*, the time necessary to click the down button, as 0.2 seconds, and the hyperlink tracing operation, *i.e.*, the time necessary for the user to click a hyperlink and have the contents of the linked page displayed on the mobile phone (including the communication time with the Web server), as 5.0 seconds. These values are defined empirically based on analysis of actual mobile phone usage. The estimated time necessary for users to browse target contents on their mobile phones is calculated for all selected target content elements, for both our system and the Google Wireless Transcoder.

6.2.3 Results

Evaluation results are shown in Figure 10. In Figure 10, the horizontal axis expresses the position of the target content element, and the vertical axis is the average estimated time necessary for the target content element to be displayed on the screen of the mobile phone. The results in Figure 10 indicate that, for "Bottom" target content elements, our system is four times more efficient than the Google Wireless Transcoder, and twice as efficient for "Middle" elements. These results prove that the proposed method significantly improves the usability of Web browsing on the mobile phone.

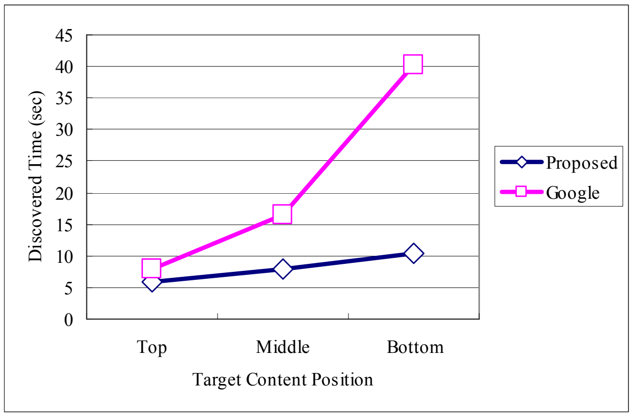
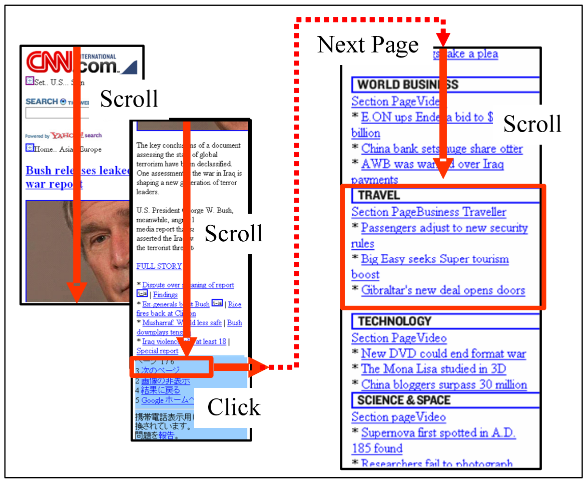
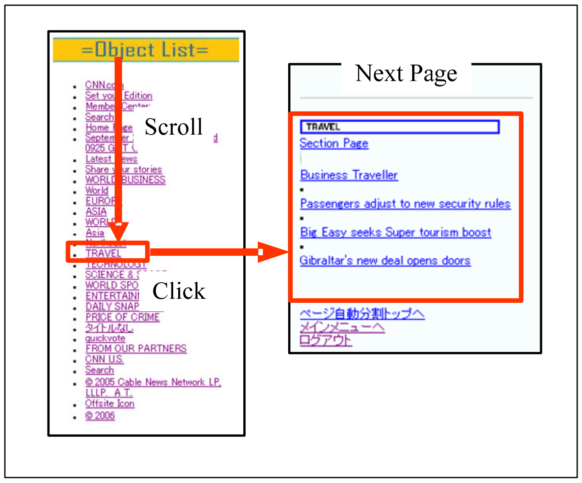


Figure 10. Usability Evaluation Results

Furthermore, we show a comparison of the usability of the two systems, by illustrating the actual user operation in a simulated task, i.e., the browsing of the "TRAVEL" category of the CNN Web site[26]. The simulated user operations for the Google Wireless Transcoder and our system are presented in Figure 11. As clear from this Figure, the amount of user operation is far fewer for our proposed method compared to Google's, indicating that our system is capable of providing an extremely efficient interface for Web page browsing on the mobile phone.



Google Wireless Transcoder


=Object List=
Scroll
Click
Next Page
TRAVEL
Section Page
Business Traveller
Passengers adjust to new security rules
Big Easy seeks Super tourism boost
Gibraltar's new deal opens doors

Our Proposed Method

Figure 11. Comparing the Usability for a Task

6.3 Performance Evaluation

In order to evaluate the performance of the proposed method, we measure the processing time on the proxy server of the proposed method.

6.3.1 Conditions

We prepared a server with 3.4 GHz CPU and 2GB memory, and select four sites from Table 1. We measured the processing time of the major processes, which consist of (a)analyzing request message, (b)parsing HTML, (c)rescaling images, (d)extracting tag depth, (e)segmentation, and (f)rebuilding XHTMLs.

6.3.2 Results

Evaluation results are shown in Table 5. The results in Table 5 indicate that the majority of the processing time is consumed for processes (b) and (c), but the overall processing time is less than 2 seconds. This processing time can be considered reasonable, since, in a practical environment, the communication time necessary to display the contents of the Web page on the mobile phone (approximately 10 to 15 seconds) is significantly more redundant than the processing time listed on Table 5.

Table 5. Performance Evaluation Results (msec)

	(a)	(b)	(c)	(d)	(e)	(f)	Total
JP-1	49.5	921.8	379.4	11.1	211.2	3.1	1576.1
JP-4	48.7	857.5	743.8	6.3	180.7	1.8	1838.6
JP-6	47.0	697.9	826.0	72.6	135.1	1.8	1780.3
JP-14	44.8	280.6	216.5	11.2	61.1	1.8	616.0

7. CONCLUSION

In this paper, we have proposed and evaluated a robust Web page segmentation method, which divides a Web page into small segments based on content-distance and Web page layout information. The proposed method is capable of solving known problems in conventional method based only on content-distance, such as the difference between the calculated content-distance and the intuitional distance between content elements of the Web page, and its incapability to handle separate Web page components such as headers, footers, and so on. Results of evaluation experiments to compare the segmentation accuracy of the proposed method with conventional methods show that a maximum 0.61 improvement in F-measure was achieved, proving that the proposed method improves Web page segmentation accuracy. Furthermore, we have implemented the proposed method for Web browsing on mobile phones, and estimated the time necessary for simulated users to browse information on mobile terminals. Comparison with existing Web page transcoders for mobile phones has shown that our method is significantly more efficient than existing methods, proving that our method is effective to improve the overall usability of mobile phone Web browsers.

8. REFERENCES

[1] Gen Hattori, Kazunori Matsumoto, Fumiaki Sugaya. Auto Web Page Distilling Scheme Based on Content-Distance Using Relative Tag Hierarchy. DBSJ Letters, Vol.4, No.1, 2005. (in Japanese)

[2] Gen Hattori, Kazunori Matsumoto, Fumiaki Sugaya. Dynamic Segmentation of a Web Page Based on Standard Deviation of Content-Distances. IPSJ Transaction of Database, Vol.47, No.SIG8, pp. 81-89, 2006. (in Japanese)

[3] Seiji Yamada, Yuki Nakai. Monitoring Partial Update of Web Pages by Interactive Relational Learning. JSAI Technical Papers of Active Mining, Vol. 17, No. 5, pp.614-621, 2002. (in Japanese)

[4] Small Screen Rendering (Opera Software ASA). http://www.opera.com/products/mobile/smallscreen/.

[5] Yu Chen, Wei-Ying Ma, Hong-Jiang Zhang. Improving Web Browsing on Small Devices Based on Table Classification. The Twelfth International World Wide Web Conference, 20-24, May 2003.

[6] Hidetaka Masuda, Shuichi Tsukamoto, Saisuke Yasutomi, and Hiroshi Nakagawa. Recognition of HTML Table Structure. The First International Joint Conference on Natural Language Processing (IJCNLP-04), pp.183-188, 2004.

[7] George Buchanan, Sarah Farrant, Matt Jones, and Harold Thimbleby. Improving Mobile Internet Usability. Proc. of 10th International World Wide Web Conference, Hong Kong, China, 2001.

[8] Jones, M., Buchanan, G., Thimbleby, H. Sorting out Searching on Small Screen Devices. Conference on Mobile HCI, 2002.

[9] O. Buyukkokten, H. Garcia-Molina, and A. Paepcke. Seeing the whole in parts: Text summarization for web browsing on handheld devices. Proc. of 10th International World Wide Web Conference, 2001.

[10] O. Buyukkokten, H. Garcia-Molina, A. Paepcke, and T. Winograd. Power browser: Efficient web browsing for PDAs. Proc. of Human-Computer Interaction Conference 2000, 2000.

[11] Y. Chen, W. Ma, and H. Zhang. Detecting web page structure for adaptive viewing on small form factor devices. In Proc. World Wide Web Conference 2003, 2003.

[12] T. Maekawa, T. Hara, and S. Nishio. A Collaborative Web Browsing System for Multiple Mobile Users. Proc. of IEEE International Conference on Pervasive Computing and Communications (PerCom 2006), pp. 22-35, 2006.

[13] Baluja, S. Browsing on small screens: recasting web-page segmentation into an efficient machine learning framework. Proc. of the International Conference on World Wide Web (WWW '06), pp.33-42, 2006.

[14] N. Milic-Frayling, R. Sommerer. SmartView: Enhanced Document Viewer for Mobile Devices. MSR-TR-2002-114, 2002.

[15] N. Milic-Frayling, R. Sommerer, K. Rodden, and A. F. Blackwell. SearchMobil: Web Viewing and Search for Mobile Devices. Proc. of the International Conference on World Wide Web (WWW '03), 2003.

[16] Anderson, C.R., Domingos, P., and Weld, D.S. Personalizing web sites for mobile users. Proc. of 10th International World Wide Web Conference, 2001.

[17] T. Maekawa, T. Hara, and S. Nishio. Content Description and Partitioning Methods for Collaborative Browsing by Multiple Mobile Users. Proc. of International Workshop on Mobility in Databases and Distributed Systems (MDDS 2005), pp. 1068-1072, 2005.

[18] Wobbrock, J., Forlizzi, J., Hudson, S., Myers, B. WebThumb: interaction techniques for small-screen browsers. In Proc. UIST '02, pp. 205--208, 2002.

[19] am H., Baudisch P. Summary Thumbnails: Readable Overviews for Small Screen Web Browsers. Proc. ACM CHI 2005, p. 681-690, 2005.

[20] Patrick Baudisch, Xing Xie, Chong Wang, Wei-Ying Ma, Collapse-to-Zoom: Viewing Web Pages on Small Screen Devices by Interactively Removing Irrelevant Content, 17th Annual ACM Symposium on User Interface Software and Technology (UIST 2004), 2004.

[21] HTML Tidy Library Project. http://tidy.sourceforge.net/.

[22] Yahoo! Business News. http://news.yahoo.com/i/749.

[23] In-Stat. http://www.instat.com/.

[24] Alexa. http://www.alexa.com/.

[25] Google Wireless Transcoder. http://www.google.com/xhtml/.

[26] CNN. http://www.cnn.com/.