Yet Another Hybrid Segmentation Tool

Andrés Sanoja
LIP6
Université Pierre et Marie Curie
Paris, France
andres.sanoja@lip6.fr

Stephane Gançarski
LIP6
Université Pierre et Marie Curie
Paris, France
stephane.gancarski@lip6.fr

ABSTRACT

In this paper[1] we present an overview of a prototype we are developing for in the context of web archives (page comparison, crawling and information retrieval). It analyses pages based on their DOM tree information and their visual rendering. This tool implements a modified version of VIPS with the aim of enhancing the precision of visual block extraction and the hierarchy construction. First, the visual rendering of a page, produced by several browsers, is segmented into rectangular blocks. Then, the extracted blocks are analysed looking for visual overlaps, which are analysed using a adapted version of the XY-Cut algorithm and resolve the overlap. As a result we may have different shapes of blocks, rectangular and non-rectangular blocks. Finally, the visual block tree, representing the layout of the page is analysed in order to have a more coherent layout disposition.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

Algorithms

Keywords

digital preservation, web archiving, web page segmentation

1. INTRODUCTION

Web pages have become one of the most important medium of nowadays information. Preserving them for future generations include harvesting them, but also develop new ways of storing, retrieving and manipulating them. This requires to process Web pages as close as possible from the way users interact with them.

[1]This work was partially supported by the SCAPE Project. The SCAPE project is co-funded by the European Union under FP7 ICT-2009.4.1 (Grant Agreement number 270137)

For users, the visual aspects of a Web page determine the way they interact with it. Users identify the objects or segments that compose the page, such as header, footer and content navigation; their characteristics, such as colour and size, as well. The underlying complexity is not relevant, for users. They don't think in terms of HTML tags or CCS properties to identify them. However, web browsers need some mechanism in order to precisely render what a user expects. We agree with Yelisada [11] and Pnueli [7] that sometimes the source code is not enough to be able to reproduce the user perspective, the objects that compose the page and the structure among them. Moreover, screenshots of the page give another perspective of the web page where the underlying structure is not reachable but give some other insights of the page, such as a layout, shapes, disposition, among others. Thus, to identify these segments, seems a very good idea to have both aspects in mind.

Web Page segmentation is a technique used for dividing a web page into particular parts, not overlapping, called blocks. These blocks can be annotated with their importance, their type (content, advertising, ...), among others. A hierarchical representation can be obtained to denote the organization of the blocks within the document. Block information is quite useful. For instance, in Web IR, regards blocks as the basic information unit of web pages and take advantages of the semantic coherence of contents within each block [3]. Block information could be used to accomplish several kinds of task, such as building web search indexes [2], web archiving [8], improve the browsing on small screen devices [1], and so forth.

A lot of research has been led in the area of page segmentation. Cai [4] proposed the VIPS algorithm that uses rule based heuristics to segment the visual layout of a web page. Baluja [1] recast web page segmentation into machine learning framework using decision tree. Xiang et all [10] define some common patterns beforehand, and then search them in the layout tree. Chakrabarti et all [5] deal with web page segmentation from a graph-theoretic perspective. Zou et all [12] have proposed a simpler version of VIPS using the recursive X-Y cut algorithm [6] for on-line medical journal papers analysis. The latter solve the overlapping blocks problem separating them in children zones, only for *table* tags with *align* property set. Among all those approaches, the one of [12] seems the most promising and we follow a similar approach. However, as they model blocks as rectangles, their method leads to create several blocks for an

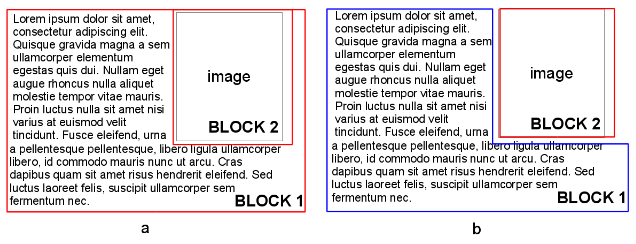


Figure 1: Overlapping blocks problem

information unit, while our method would create only one, possibly non rectangle, block.

VIPS is a recursive algorithm which accepts a DOM tree and some visual cue (background, color, size, ...) as input and produces a hierarchy, closely related to how a user would perceive the page. The algorithm is performed in three phases: Visual Block Extraction, Visual Separator Detection and Content Structure Construction. At a glance, the DOM tree is analysed beginning at the root node. Each children node can be either divided or extracted following the heuristic rules. The extracted nodes, as blocks, are put in a pool. The separators between blocks are detected based on the visual layout. A weight is assigned to each separator, depending on certain conditions. Then, the less weighted blocks are merged to form more coherent blocks. Each node corresponds to a block in the page, and has a value to indicate the Degree of Coherence (DoC). The process continue recursively until the DoC of the leaf node meets the predefined threshold.

After segmenting a page, one may think that determining to which block a DOM node corresponds is straightforward: compare the boundaries of a block (left, top, bottom and right) and the position of the node. However, while designing and building the segmentation algorithm we have found that in some cases this is not entirely true. Sometimes, the rectangle shape is not well suited for blocks, because overlapping may occur, making the model ambiguous. Figure 1a shows an example for a tag DIV with textual content and an image with the attribute *align* set. VIPS has detected two sibling blocks. We cannot tell exactly if the tag *img* is contained in block 1 or 2. From the source code point of view there is no visible problem: there are two nodes and therefore two independent blocks (assuming the heuristics rules are met). However if we look at the screenshot, we notice that the text flows along with the image and an overlap occurs. The visual representation does not correspond to the underlying source code and thus it is not obvious to know to which block the tag **img** corresponds to. Figure 1b depicts how with we think this ambiguity could be resolved: block 1 with a non rectangular shape will fix the overlap problem.

Another issue raised by VIPS is that the position of the blocks in the hierarchy do not necessarily correspond with the layout flow of the document. Figure 2 shows an example of this situation. We think that the hierarchy should be self-contained if we observe it, in other words, if we can infer the structure or the layout of a page from it. For example a block representing a header should be ranked first and then

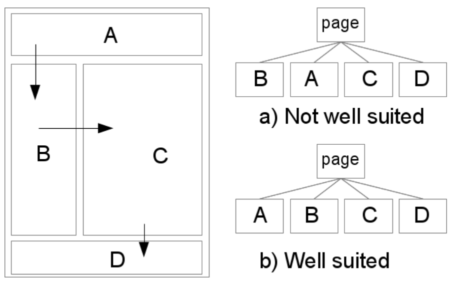


Figure 2: Layout flow problem

the navigation block and later the content block. This issue is closely related to the CSS Regions proposed by W3C [9], which provides an advanced content flow mechanism, which can be combined with positioning schemes. We suggest the deep study of those schemes in order to use them in the segmentation process.

Ambiguities in the segmentation process affects the precision of the subsequent activities. For instance, in Web (or Web archives) search engines, block-based indexes can fix this ambiguity to get a better precision. A similar problem was reported by Yang [10] who use two approaches to web page segmentation (VIPS and Gestalt-based one) to classify the roles of images as block features. They need to have a set of no overlapped blocks, and they present a method for achieved it.

In this paper we describe a prototype for segmenting web pages in the framework of the SCAPE project, particularly in the web archive context. First, the segmentation algorithm is introduced. It follows a hybrid approach for enhancing the precision of page segmentation. It uses a modified version of VIPS [4] for DOM segmentation and adapted version of the XY-Cut algorithm[6] for image segmentation. The rest of this paper is organized as follow. In section 2 we describe the hybrid approach which we propose. Section 3 introduce the prototype. Section 4 concludes.

2. HYBRID APPROACH FOR WEB PAGE SEGMENTATION

In this section we describe our hybrid segmentation approach. First we describe the insights considered to enhance the Visual Block Extraction phase of VIPS in order to get a more precise segmentation, and to avoid the overlaps. Then, we describe the steps to enhance the hierarchy of VIPS, it determines the location of blocks according to the position in the layout. Figure 3 depicts the segmentation algorithm, based on the original algorithm with enhancements.

2.1 Decorated HTML and Screenshot

The goal of this first step is to prepare the data needed for segmentation. Two input files are required: a Decorated HTML and a screenshot file. The decorated HTML is the original source document without tags that give no information for the segmentation (script, style and comments tags). For each remaining tags, extra attributes are added to denote the visual cues (left, top, width, height, background color, font style, margin-width, margin-height and background property). This self-documented document can

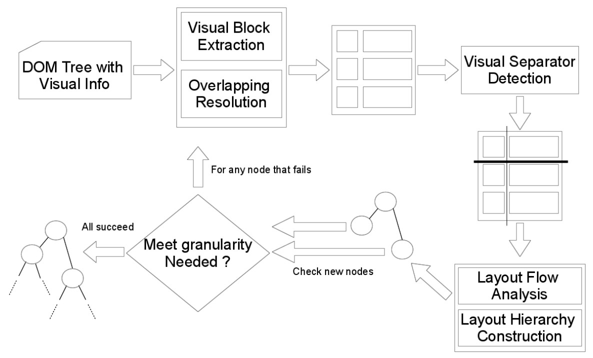


Figure 3: Segmentation Algorithm: VIPS Algorithm with enhancements

be stored in a repository or file system preserving the state of the rendering engine for later use.

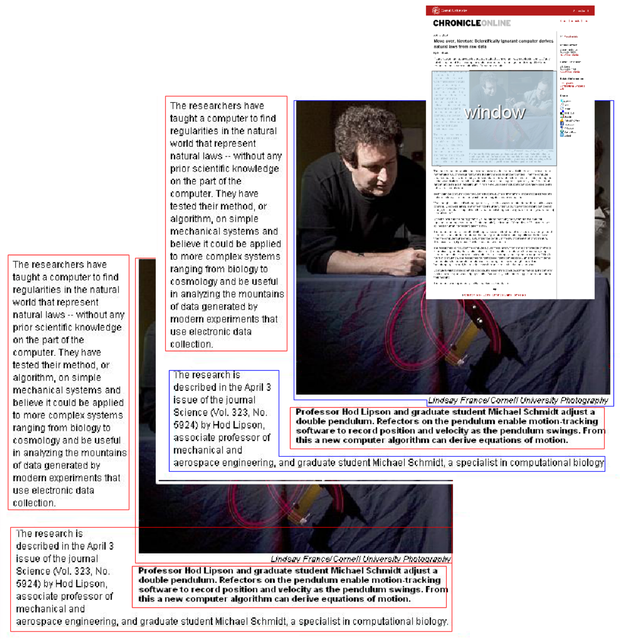
2.2 Visual Block Extraction and Overlapping Resolution

We apply the segmentation algorithm over the decorated HTML and obtain the visual blocks and the hierarchy. Based on the position of each block we detect possibles overlaps. In this case we may have two or more blocks. We calculate the area held by these blocks and define a temporary window. Within this window, a XY-Cut algorithm is applied. For each new non-rectangular block, we determine if it is contained in the original rectangular block. If it is the case we remove the rectangular block and replace it by the equivalent non-rectangular polygon. Of course, because the algorithm is recursive, the window size directly depends on the parent DOM node area. Figure 4 shows the result of applying the DOM segmentation to a page with overlapping blocks. Figure 4 depicts also the window defined by the overlapping blocks area: image and caption, image description and the left paragraph. The top paragraph and the image caption remain unchanged because the result of the analysis is the same rectangle block. For the others (image and bottom-left paragraph) the non-rectangular blocks detected are included in the original blocks area, and therefore are removed. Other cases are not described here due to limitation of space in this paper, but it is worthy to highlight that it is possible that two rectangular blocks in a window would be replaced by a non-rectangular one. Assume by example that we have two tags DIV representing two different blocks. Using DOM segmentation, they are detected as two blocks. If they are included in a window they could be merged into one block, if we can infer from the geometry detected by the image segmentation that they are very close.

2.3 Layout Hierarchy Construction with Block Correspondence

Figure 2 shows an example of a hierarchy that does not correspond to the layout flow of the document. Figure 2a is an example of segmenting a wikipedia web page [2] with the VIPS algorithm. The resulting hierarchy satisfies the relationship parent/children. However the siblings are not in the same order as in the layout flow. In other words, using

[2]http://es.wikipedia.org/wiki/Acteon


CHRONICLEONLINE
Move over, Newton: Scientifically ignorant computer derives natural laws from raw data
window

The researchers have taught a computer to find regularities in the natural world that represent natural laws -- without any prior scientific knowledge on the part of the computer. They have tested their method, or algorithm, on simple mechanical systems and believe it could be applied to more complex systems ranging from biology to cosmology and be useful in analyzing the mountains of data generated by modern experiments that use electronic data collection.

The research is described in the April 3 issue of the journal Science (Vol. 323, No. 5924) by Hod Lipson, associate professor of mechanical and aerospace engineering, and graduate student Michael Schmidt, a specialist in computational biology.

Lindsay France/Cornell University Photography

Professor Hod Lipson and graduate student Michael Schmidt adjust a double pendulum. Refectors on the pendulum enable motion-tracking software to record position and velocity as the pendulum swings. From this a new computer algorithm can derive equations of motion.

The researchers have taught a computer to find regularities in the natural world that represent natural laws -- without any prior scientific knowledge on the part of the computer. They have tested their method, or algorithm, on simple mechanical systems and believe it could be applied to more complex systems ranging from biology to cosmology and be useful in analyzing the mountains of data generated by modern experiments that use electronic data collection.

The research is described in the April 3 issue of the journal Science (Vol. 323, No. 5924) by Hod Lipson, associate professor of mechanical and aerospace engineering, and graduate student Michael Schmidt, a specialist in computational biology.

Lindsay France/Cornell University Photography

Professor Hod Lipson and graduate student Michael Schmidt adjust a double pendulum. Refectors on the pendulum enable motion-tracking software to record position and velocity as the pendulum swings. From this a new computer algorithm can derive equations of motion.

Figure 4: Overlapping blocks example and resolution using image analysis

VIPS algorithm does not allow to infer the real structure of the layout. This is an important issue when building an index or when comparing page versions to improve the crawling in Web archives. For each block in the hierarchy we evaluate the absolute position and the area that it occupies. It is compared with the others siblings blocks, performing a sort based on this criteria. This continues recursively for the children of each block.

3. PROTOTYPE DESCRIPTION

In this section we describe the YAHSET functional prototype. It is an open-source platform-independent web page segmentation tool designed to facilitate the web page analysis through the hybrid web page segmentation approach. It is composed of a module for managing the data visualization and user interaction, a module for preprocessing input files and another for page segmentation. Figure 5 shows the overview of the overall process denoting the web page preprocessing phase, page segmentation and the visualizer.

3.1 Page Preprocessing

The goal of this phase is to have all the data needed for segmentation off-line: that is the decorated HTML file and image file with screenshot. We use a set of different browsers which are requested to download and render a web page and generate a screenshot. The reason to use several browsers is that we would like to evaluate the segmentation results. The screen-shots and the decorated HTML are processed using Selenium WebDriver and Selenium Server. For each URL given a screen-shot and decorated HTML are obtained following the way rules are interpreted on the browsers. To keep results homogeneous, each browser width is resized to 1024 pixels. A set of JavaScript scripts are injected to the browsers in order to get the HTML document with all the visual cues includes as attributes and the screen-shots. Fig-

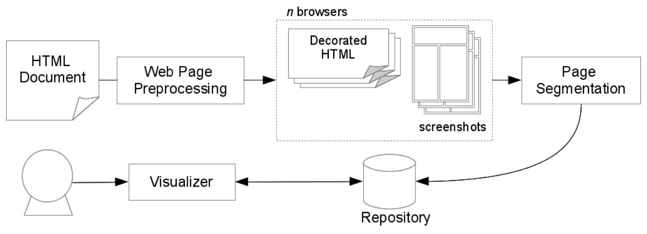


Figure 5: Prototype Process

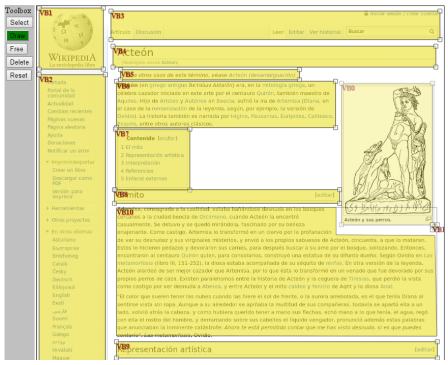


Figure 6: Data Visualization and User Interaction

ure 6 we can see the GUI for visualizing the segmentation results and also for getting feed back from users.

3.2 Data Visualization and User Interaction

Data visualization and user interaction are done though a web application. Users give the URL of the pages they want to process. The can use the basic web resource operations (add, modify, delete and list). A page for visualizing the segmentation results in a HTML5 compliant browser is available. The users can interact with the visual blocks detected: move them, tune them, add and delete new ones. This feedback information is stored for later use.

3.3 Page Processing

This module is the core of our prototype. It includes the algorithms for web page segmentation, and the page preprocessing routines. For each document (decorated HTML and image) the segmentation algorithm is used, following the procedure described in Section 2. The results are saved in a database, for further utilization. The segmentation algorithm is implemented in Ruby 1.9.2 using the libraries for HTML/XML manipulation Hpricot. For image manipulation a modified version of the XY-Cut algorithm is used.

4. CONCLUSION AND FUTURE WORK

In this paper we described a prototype for Web Page Segmentation. This tool aims to be part of the tools developed in the framework of the SCAPE project. Our paper points out the issues of efficiently segmenting web pages and improving the quality and the relevance of the segmentation. To address this issue, we propose an hybrid approach that, in contrast to previous approaches, use both the DOM analysis and image segmentation (usually they are used apart). Partial results show some extra overhead in the process but also exhibit (in some way) more coherent results.

As a future work, we look forward to improve more significantly the precision of the segmentation and include those insights in all aspects of an archive, indexes for example. Another on-going work is to experiment our algorithm with a bigger set of web pages, not only test cases. We intend to use the feedback from the users as a input to a Knowledge Base which would help to improve the Visual Block Extraction Phase, therefore the quality of the archive itself. Beside this, we are would like to test the hybrid approach to include information that usually with DOM approaches only are not reach, such as flash movies.

5. REFERENCES

[1] S. Baluja. Browsing on small screens: recasting web-page segmentation into an efficient machine learning framework. In *Proc. of the 15th international conference on World Wide Web*, WWW '06, pages 33–42, New York, NY, USA, 2006. ACM.

[2] E. Bruno, N. Faessel, H. Glotin, J. Le Maitre, and M. Scholl. Indexing by permeability in block structured web pages. In *Proc. of the 9th ACM symposium on Document engineering*, DocEng '09, pages 70–73, New York, NY, USA, 2009. ACM.

[3] E. Bruno, N. Faessel, J. L. Maitre, and M. Scholl. Blockweb: An ir model for block structured web pages. *Content-Based Multimedia Indexing, International Workshop on*, 0:219–224, 2009.

[4] D. Cai, S. Yu, J.-R. Wen, and W.-Y. Ma. Extracting content structure for web pages based on visual representation. In *Fifth Asia Pacific Web Conference (APWeb'03)*, 2003.

[5] D. Chakrabarti, R. Kumar, and K. Punera. A graph-theoretic approach to webpage segmentation. In *Proc. of the 17th international conference on World Wide Web*, WWW '08, pages 377–386, New York, NY, USA, 2008. ACM.

[6] B. Kruatrachue, N. Moongfangklang, and K. Siriboon. Fast document segmentation using contour and x-y cut technique. In C. Ardil, editor, *WEC (5)*, pages 27–29. Enformatika, Çanakkale, Turkey, 2005.

[7] A. Pnueli, R. Bergman, S. Schein, and O. Barkol. Web page layout via visual segmentation, 2010.

[8] M. B. Saad and S. Gançarski. Using visual pages analysis for optimizing web archiving. In *Proc. of the 2010 EDBT/ICDT Workshops*, EDBT '10, pages 43:1–43:7, New York, NY, USA, 2010. ACM.

[9] W3C. Css regions http://dev.w3.org/csswg/css3-regions/, 2012.

[10] X. Yang and Y. Shi. Enhanced gestalt theory guided web page segmentation for mobile browsing. In *Proc. of the International Joint Conference on Web Intelligence and Intelligent Agent Technology*, pages 46–49, Washington, DC, USA, 2009.

[11] Y. Yesilada. Web page segmentation: A review. Technical report, Middle East Technical University Northern Cyprus Campus, March 2011.

[12] J. Zou, D. Le, and G. Thoma. Combining dom tree and geometric layout analysis for online medical journal article segmentation. In *Digital Libraries, 2006. JCDL '06. Proceedings of the 6th ACM/IEEE-CS Joint Conference on*, pages 119 –128, june 2006.