Web Page Layout Via Visual Segmentation

Ayelet Pnueli, Ruth Bergman, Sagi Schein, Omer Barkol

HP Laboratories
HPL-2009-160

Keyword(s):
Layout understanding, Layout analysis, Web page segmentation, HTML, DOM

Abstract:

Web page segmentation is required for any application that observes, manipulates, interacts, summarizes or does anything with web content or web services. Although segmentation is a non-trivial task, until recently it could be performed reasonably by analyzing the HTML structure. Today, the dynamic content of web pages does not fit the assumptions made by those algorithms. The HTML structure does not contain enough information to extract the important regions. Yet, visually, the page itself remains understandable to the human user. Thus, we believe it contains all the information that is needed to understand its content. We propose adding methods of computer vision for the analysis of the page. When the HTML does not contain the needed object hierarchy information, one may use the visual information. Moreover, visual segmentation allows us to correct the HTML structure or to simplify its hierarchy which in many cases is not semantic. We perform top-down segmentation, yielding first the large scale layout of the page, down to the required degree of detail.

External Posting Date: July 21, 2009 [Fulltext] Approved for External Publication
Internal Posting Date: July 21, 2009 [Fulltext]

Web Page Layout Via Visual Segmentation

Ayelet Pnueli
HP Labs
Technion City
Haifa, Israel
++97248231237
ayelet.pnueli@hp.com

Ruth Bergman
HP Labs
Technion City
Haifa, Israel
++97248231237
ruth.bergman@hp.com

Sagi Schein
HP Labs
Technion City
Haifa, Israel
++97248231237
sagi.schein@hp.com

Omer Barkol
HP Labs
Technion City
Haifa, Israel
++97248231237
omer.barkol@hp.com

Abstract

Web page segmentation is required for any application that observes, manipulates, interacts, summarizes or does anything with web content or web services. Although segmentation is a non-trivial task, until recently it could be performed reasonably by analyzing the HTML structure. Today, the dynamic content of web pages does not fit the assumptions made by those algorithms. The HTML structure does not contain enough information to extract the important regions. Yet, visually, the page itself remains understandable to the human user. Thus, we believe it contains all the information that is needed to understand its content. We propose adding methods of computer vision for the analysis of the page. When the HTML does not contain the needed object hierarchy information, one may use the visual information. Moreover, visual segmentation allows us to correct the HTML structure or to simplify its hierarchy which in many cases is not semantic. We perform top-down segmentation, yielding first the large scale layout of the page, down to the required degree of detail.

Categories and Subject Descriptors

I.4 [**IMAGE PROCESSING AND COMPUTER VISION]:** I.4.6 Segmentation, I.4.9 Applications, I.5.2 Design Methodology,.
I.7 [**DOCUMENT AND TEXT PROCESSING**]: I.7.5 Document Capture/
J. [**Computer Applications**].

General Terms

Algorithms, Theory,

Keywords

Layout understanding, Layout analysis, Web page segmentation, HTML, DOM

Introduction

As the amount of information and services available via the web increases, the use of web for accessing information, shopping, and communicating is increasing, and, simultaneously, web-centric business models are evolving. These changes have resulted in a more sophisticated presentation of content on the web. A web page typically displays a number of different messages to the user, which are usually visually distinct. For example, a web page might contain advertisements and links to other relevant pages in addition to the main content of the page. Thus, an application that intends to re-use content on the web, such as a search engine or a web-to-print application, needs to identify the regions of the page that contain distinct information. Other applications, such as web automation applications, need to identify user-interaction (UI) components, such as buttons and links. Until recently, it has been possible to identify these regions and components from the HTML code that generated the page. The recent trend toward dynamic web technologies implies that the HTML not longer contains sufficient information on the content of the page. Sometimes it contains almost no information, e.g., in the case of flash presentations. To the user, on the other hand, the page remains perfectly understandable. In other words, the page itself contains all the information that is needed to understand its content. We therefore conjecture that by analyzing an image of the page it is possible to extract the needed information. This paper describes an algorithm that segments a web page recursively to segment the layout of the page and the UI components using the page's rendered image.

For our purposes, a segment on the web page is a region that a user would identify as distinct from the rest of the page in some way. Prior approaches to the web-page segmentation problem [Cai03, ChakrabartiEtAl2008] view a segment as corresponding to a fragment of HTML. These solutions analyze the HTML Document Object Model (DOM), extract information about the appearance of objects on the page, and group HTML objects based on this information. While these solutions work well in traditional HTML pages, they fail in dynamic HTML applications. Likewise, web automation tools, such as HP Quick Test Pro, Selenium and Chickenfoot often fail in dynamic HTML applications.

In dynamic HTML pages, it is often the case that the object hierarchy is available, but it does not describe the layout and components semantically, e.g., a button is made up of three distinct elements each of which is comprised of a DIV element containing an image element. The behavior of the button is obtained by javascript code. Similarly the layout of the page is masked by superfluous elements. This cumbersome HTML structure arises because HTML pages are produced automatically from toolkits, e.g., the Google Web Toolkit (GWT). Other scenarios, such as pages containing Adobe Flash objects, there is virtually no information about the content in the HTML DOM. Thus, we sometimes have to understand the relevant segments and objects in the application using screen images only, or screen images combined with partial information.

Figure 1 - Highest Level Layout

In this paper, we propose a top-down algorithm for obtaining a layout of a web page from its rendered image. The layout is a segmentation of the page to meaningful, large, objects. When we apply the layout algorithm successively, we divide the page to smaller and smaller components, according to the natural visual hierarchy. At the lowest level it segments multimedia content, such as text, images, and videos, and UI components, such as buttons and edit boxes.

Previous Work

There is a large body of prior solutions on segmentation of natural images. There are many good algorithms for general segmentations, including watershed [Vincent91], mean-shift [Comaniciu02], region growing methods [Zhu95], spectral graph methods, such as normalized cuts [Shi97], and self-similarity methods [Bagon08]. Other algorithms segment a specific object type, e.g., faces, sky, and foliage [Bergman09]. Segmentation algorithms depend heavily on the statistics of natural images, or natural images of a given type. They look for edges or large gradients in the image and attempt to combine the edges into coherent blocks.

GUI images are very different from natural images. The variety is limited, and we can make the assumption that the GUI is designed to be easy to understand. This assumption means both that the objects are categorized to well known groups, so that when a person gets to a new application his prior experience will enable him to understand it quickly, and that graphically the objects are easily detectable (e.g., because they have sharp edges or color transitions).

Prior work is found on layout segmentation for Web applications. In this work, the HTML DOM is assumed to be available. The layout of the document is completely described by the DOM, and there are several good solutions for extracting the layout, most notable the VIPS algorithm [Cai03]. The DOM information, however, does not always results in the correct semantic inclusion relationship. In addition, this solution fails if the DOM does not contain all the GUI information. For example, if the GUI was formed using one flash object the DOM contains only one element.

Our Work Via an Example

In Figure 1 we see the image of "Travelocity" website, with the highest level layout drawn in red. This layout was found using edge analysis. Many times, the main objects are

Figure 2 - Second Level of the Layout (in Blue)

outlined so that there is a border between them and the area around them. This allows the user, and also a computer program, to easily find the main components of the application.

To get this layout, the algorithm uses edge analysis on the GUI image (or a transformation of it), looks for long edges directed in the horizontal and vertical directions, and then

Figure 3 - Third Level of the Layout (in Green)

selects the rectangles that do not lie within any other rectangle in the GUI image. After this stage, the algorithm seeks for areas containing information, and groups them into distinct layout elements. This technique gives us the high level layout, thereby segmenting the page to its main components. In the Travelocity example, the highest level layout separates the main content from the title and bottom navigation area. Note that all we use is visual information – at this stage we do not apply any semantic analysis to group or ungroup elements.

Next, the algorithm finds the next level of layout, by recursively computing the outer-most rectangles within each layout element. The second level of recursion for our example is shown in Figure 2 in blue. This recursive process may continue until down to the level of individual elements, which may be text areas, images, videos, buttons, edit boxes, etc. Alternatively, the user may terminate the recursion earlier, depending on the application at hand. The third level of recursion is shown in green in Figure 3.

As we go deeper in the hierarchy, this task becomes more difficult because the objects we separate become smaller. Thus, the edges are denser and tend to merge. To solve this problem we use additional properties of the image, including color, brightness and text detection, to improve the segmentation.

Additionally, to accurately segment GUI components, such as radio buttons and check boxes, which are not typically rectangular, we use custom detectors. These detectors are tailored to the distinct appearance of these specialized components, via a template matching approach. The result of applying radio buttons and text detectors is shown on Figure 4. Radio buttons and check boxes are marked in blue, and text in magenta. Note that this is the highest level of layout one may wish to achieve.

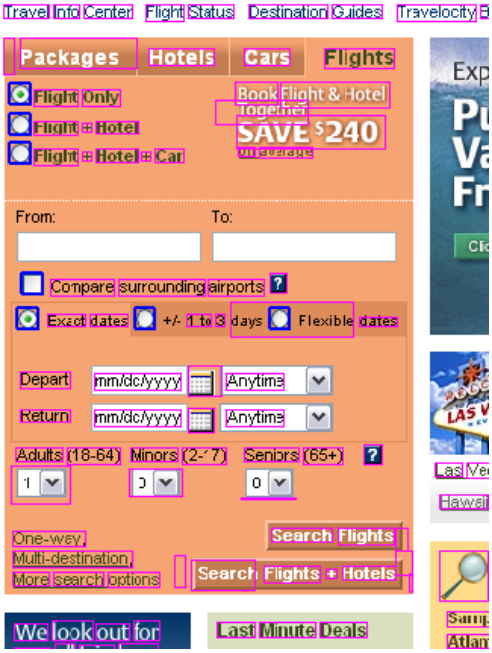

Figure 4 - Radio Buttons Segmentation

Following the text detection, we apply OCR, which can also give us information about the meaning of a layout object. The text information is important for later semantic analysis of the page content.

Future Work

In the future, we intend to combine the DOM information with the GUI image information, which we expect to improve the segmentation over using one type of information. Using image analysis techniques we will find areas or details where the DOM information is not complete, and use the information from the image to correct it. In areas where there is no DOM information (e.g., a flash object), we would use only the information from the image. This approach yields additional semantic information for each object, in addition to better segmentation.

References

[Cai03] Deng Cai, Shipeng Yu, Ji-Rong Wen and Wei-Ying Ma, “VIPS: a Vision-based Page Segmentation Algorithm” MSR-TR-2003-79

[ChakrabartiEtAl2008] Deepayan Chakrabarti, Ravi Kumar and Kunal Punera, ”A Graph-Theoretic Approach to Webpage Segmentation” WWW 2008, Beijing, China

[Comaniciu02] Dorin Comaniciu and Peter Meer, “Mean Shift: A robust approach Toward Feature Space Analysis”, Transactions on Pattern Analysis and Machine Vision, Vol 24 No 5 2002.

[Shi97] Jianbo Shi, Jitendra Malik, ” Normalized Cuts and Image Segmentation”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 22, No 8, Pages 888-905, 1997

[Vincent91] Lee Vincent and Pierre Soille,”Watersheds in digital spaces: An efficient algorithm based on immersion simulations”, Transactions on Pattern Analysis and Machine Vision”, Vol 13 No 6, pages 583—598, 1991.

[Zhu95] S.C. Zhu and T.S. Lee and A. Yuille, Region competition: unifying snakes, region growing and mdl for image segmentation, ICCV, vol 25, pg. 416-425, 1995.

[Bagon08] Shai Bagon and Oren Boiman and Michal Irani, What is a Good Image Segment? A unified Approach to Segment Extraction}, Computer Vision -- ECCV 2008, David Forsyth and Philip Torr and Andrew Zisserman Editors, pg. 30-44, Springer, 2008.

[Bergman09] Ruth Bergman, Hila Nachlieli and Gitit Ruckenstein, Perceptual Segmentation: Combining Image Segmentation with Object Tagging, submitted to IEEE Transactions on Image Processing, 2009.

[Selenium] Selenium web application testing system. http://seleniumhq.org/

[Chickenfoot] Chickenfoot: rewrite the web. http://groups.csail.mit.edu/uid/chickenfoot/publications.html