Information Extraction and Search in Newspaper Front Pages

Tiago Varela[2] and Sérgio Nunes[1,2]

[1] INESC TEC
[2] DEI, Faculdade de Engenharia, Universidade do Porto, Portugal
{tiago.varela, sergio.nunes}@fe.up.pt

Abstract. This study analyses existing newspaper archives and reviews the state of the art regarding page segmentation methods, text recognition in documents and web search interfaces. We also detail a new keyword extraction method that specifically targets current newspaper front pages. This method splits the newspaper front page image into rectangular regions and applies a background/foreground segmentation method that targets the contained text. Using a test bed designed specifically for this study, this method obtained an 8.62 percentage point increase in accuracy over the default keyword extraction method. We also developed a prototype content-aware newspaper archive, aptly named *Gazeta*, which allows the users to search for keywords and to view the front pages containing those particular keywords. It also graphically displays the number occurrences for the specified keywords, over a particular timespan.

1 Introduction

Digital versions of newspapers and magazines are becoming increasingly popular. There are free newspaper archives, like *Newseum*'s *Today Front Pages* [16] or *Kiosko.net* [14], that gather daily newspaper front pages but the search options are very limited, allowing only to view the latest editions and to search for issues, although only by date publication or country. Few newspaper archives are *content-aware*, allowing users to search for text content that was printed on the front page.

Extracting information from current newspapers is quite challenging, because these front pages do not have a strict layout and the information is scattered all over the page, e.g. with different fonts and font sizes, text directly over images without a background framing it, text lines wrapping contoured images. Figure 1 displays 2 newspaper issues, from daily Portuguese publications, and the diversity of their layout arrangements is clear, with different sizes between news titles and body text, and with variations in font or colour. Displaying this information to the users is also challenging. Search user interfaces often have cluttered interfaces that detract from a good user experience. Trying to pleasantly represent all the extracted information and provide a comfortable and easy way to use the interface is a difficult task.

(a) *Jornal de Notícias.* (b) *Público.*

Fig. 1. Current front pages from Portuguese daily newspapers.

SAPO has a large collection of newspaper front pages on their *Banca de Jornais SAPO* newspaper archive. Between December 29, 2011 and April 18, 2013, we were able to retrieve from it 11,728 images, corresponding to 3,842 unique issues from 36 different publications. This was the starting point for this study. With this collection it would be interesting to create a system, that could automatically extract the information from the newspapers front pages and, not only provide rich information about the content of these front pages, but also display the information in such a way that inspired the user to look for it or to explore related information.

2 Previous Work

2.1 Newspaper Archives

There are many websites that store images of newspapers and magazines. Some only keep front pages [16, 14, 20] while others keep the complete issue [3, 12, 17, 8]. Most provide only simple search capabilities, e.g. to find newspapers for a given date or to find newspapers or magazines from a specific country or state. Few provide more advanced search capabilities like searching for keywords inside a newspaper issue.

The *Today's Front Pages* section of the *Newseum* website [16] provides each day over 800 newspaper front pages, 7 of them Portuguese newspapers. There are no search capabilities, and the user can only: list the newspapers by region, view a gallery with small images of the front pages and the name of the corresponding newspaper, or view the newspaper distribution using a map.

Kiosko.net [14] provides many daily newspaper covers from various countries, including 5 Portuguese daily newspapers and 2 Portuguese sports newspapers. It does not provide searching capabilities beyond filtering by region or country.

Banca de jornais SAPO [20] is a Portuguese newspaper archive which allows access to the front pages of national and international newspapers and magazines with broad themes such as sport, financial, technology, and daily newspapers. It only allows access to the current day's front pages. It also only allows the user to search by newspaper or magazine names, or to search by theme.

Mário Soares Foundation provides access to *Diário de Lisboa Online* [3], an archive that stores all editions with full content of the Diário de Lisboa newspaper over it's circulation span, from 1921 to 1990. It only allows the user to navigate from issue to issue and inside each issue from page to page—search functionalities do not exist.

Google News website has a section about old newspapers [12] which provides listing and navigation through the full newspaper content. It has news segmentation [5], allowing the user to link directly to a particular news post from a newspaper. However it does not allow the user to search the contents of a newspaper and the global search includes results that are not exclusive to the newspaper archive.

NewspaperArchive is a paid subscription website [17] that keeps mostly newspapers from the USA, Canada and United Kingdom. It allows the user to browse newspaper pages, to easily, yet inaccurately zoom in and out onto a page and, the most interesting feature, to search a newspaper using keywords, enabling the user to find content that is present on the newspaper's news. After a user searches for a particular keyword the system highlights it on the displayed page.

The city council of Arganil provides an archive of an old weekly newspaper — *A Comarca de Arganil*, containing the first series from 1901 to 2009 [8]. There is an advanced search feature that allows the user to search by keywords, date, edition, or to provide a issue number range and view the issues available. This search by keyword allows the users to search for news content inside any of the newspapers.

2.2 Page segmentation

Google had a *Newspaper Digitization, Index and Search* project where they scanned microfilm pictures of old newspapers, and treated that information in order to index it [5]. The segmentation accuracy of this method was of approximately 90% and the OCR accuracy of around 80%. The segmentation accuracy was measured against manually annotated ground truth and the OCR accuracy by the ratio of dictionary words correctly recognized. Despite the good results described, this project worked only with older newspapers which had a simple layout with text blocks properly organized and without many pictures between them. Because of this the OCR component of the process could be performed in a earlier stage.

The ICDAR had a page segmentation competition with four editions (2003, 2005, 2007 and 2009), organized by the *PRImA* [18] research lab. The goal of

this competition was to evaluate methods of document layout analysis (page segmentation and region classification) [1].

The *BESUS* method [1, 6], which was the best overall method of the 2007 competition, used morphological operations to detect graphics and line art and extracted those. What remained on the page was then mostly text.

The *Fraunhofer Newspaper Segmenter* [2] was the best overall method for the 2009 competition. It generated binary images from the input document and detected separator lines and gutters. They then segmented the page using an hybrid method for grouping the smaller page components, guided by the overall column information given by the line separators and gutters.

2.3 News Visualization

Newsmap[3], a website created in 2004, displays an updated *Google News* feed using a *treemap* representation—the system is displayed in Fig. 2. This interesting

(a) Properly sized news.

(b) News item too big, unbalancing the *treemap*.

Fig. 2. *Newsmap* website showing a *treemap* with news.

way of representing news items uses colours to represent categories and the size of each block is given by the number of related articles, effectively giving more emphasis to stories that had more coverage [24]. It's a very effective way to have an overall view of the most important news at that moment, however sometimes the generated tree blocks are too small to contain information—this defect is visible in Fig. 2b.

Also in 2004, *Stamen*[4] created *In The News* [21], an interface that provided visual comparison of specific topics based on occurrences over time. This visualization displayed the news as coloured boxes across a vertical weekday axis. Each colour represented a news topic and the box size showed the topic's relevance for a particular day. Colour hues were used to denote if a particular topic had lost or gained importance from the previous day [25]. It also displayed *sparkline-style* graphs that provided a quick overview on each of the specified topics [21]. The

[3] http://newsmap.jp
[4] http://stamen.com/

information was initially retrieved from *Google News*, and ,later on the life of the project, from *Delicious* most popular items. Figure 3 shows a sample of the visualisation interface for the 2004 USA presidential election.

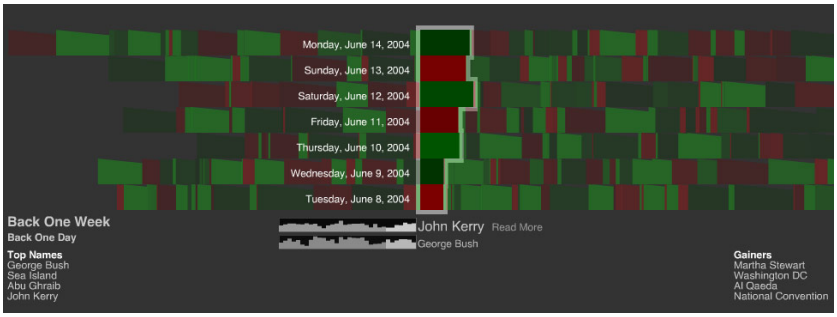


Fig. 3. *Stamen*'s *In the News* providing an overview of the 2004 USA presidential election.

3 Newspaper Extractor

The *newspaper-extractor* was developed in *C++* and used the *OpenCV 2.3.1* [22] library to process the images, and the *Tesseract 3.0.2* [23] library to perform character recognition. The *gflags 2.0* [11] library was also used to easily allow command line flags to alter the behaviour of the program.

The application starts by detecting the rectangular regions that are present in the image. This is done by finding only the extreme outer contours of the images, which are most external regions separated from other regions through a white gutter, i.e. the newspaper background colour. Figure 4 displays a newspaper front page and its outer contours.

Once the external regions are identified, they are filtered to select only the rectangular shaped ones. This filtering is a very basic one, the bounding box of the contour is obtained and if the width multiplied by the height, of that box, is approximate to the area of the contour then the region is kept, as it is a rectangular external region. Also in this step, a first pass with the *Tesseract* library is done and rectangles with less than two detected text characters are discarded from the external rectangular list of regions. This is done because the rectangular regions that we are after are the ones surrounding news item, and some of the detected rectangles correspond only to single characters that have rectangular shapes. These discarded rectangles are, for example, the ones corresponding to capital letter "I" in sans-serif fonts or the capital "D"s whose shape is, in some fonts, similar to a rectangle. Figure 5 shows the detected rectangular regions, as well as false positives that were subsequently removed.

The next step is to apply a binarization method to each of the detected rectangular regions. The default one is the FBCI text binarization method [13],

(a) Input newspaper image. (b) All extreme outer contours.

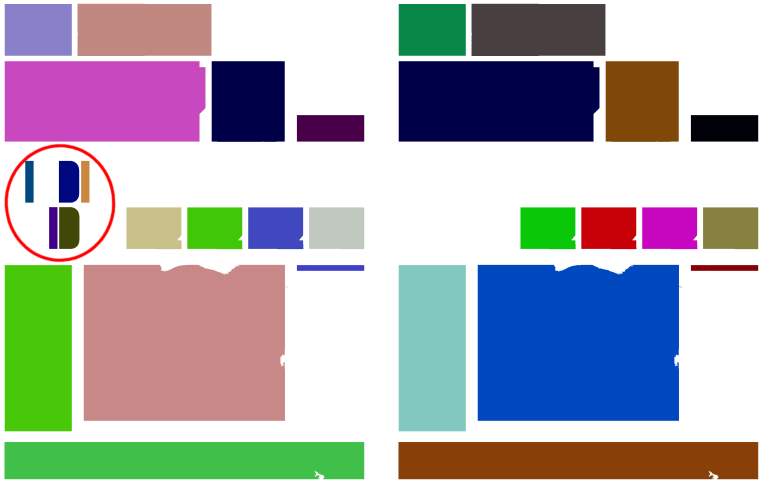
Fig. 4. Front page external contours.

but a global binarization method was also developed and can be used by passing a flag in the command line. The FBCI text binarization method has a different approach to binarization where the values of the edges from characters in the image are used to generate the correct threshold value for that particular edge. Also, this method always outputs a image where the foreground text colour as black and the background as white—this avoids further processing as that is the correct way to output text images to the *Tesseract* library.

Our implementation of the FBCI starts by using the Canny edge detection algorithm on each of the three colour channels of the input image. The thresholds used during this step are 0.66 and 1.33 times the mean channel value [26]. Then, the three resulting images are combined into one using a logical *OR* operation and the contours of that output image are archived. These contours correspond now to the text characters, pictures and dividing lines of the image, but since we want only the text characters a quick filtering must be done first. Here we get the bounding box of the contour and keep only the ones where the aspect ratio, i.e. the width divided by the height of the box, fits between 0.1 and 10, and the box area is less then 1/5th of the area from the input image [13].

Once the contours are filtered we start estimating the thresholds for each contour. Here we set the the *foreground-value* as the average colour from the border pixels for each contour, and set the *background-value* as the median value of a sample of twelve background pixels, i.e. three pixels from each of the corners of the contour's bounding box.

Now, if the foreground-value is lower than the background value, i.e. darker foreground than the background, the threshold function uses a binary operation (1), where if a pixel in the area of that contour, *src*, is higher than the calcu-



(a) Rectangular regions with false positives highlighted.

(b) Correct rectangular regions.

Fig. 5. Detecting external rectangular regions from a front page.

lated foreground-value then that same pixel in the resulting image, *bin*, is set as white—or black otherwise.

$$bin(x,y) = \begin{cases} white & \text{if } src(x,y) > \text{foreground-value} \\ black & \text{otherwise} \end{cases} \quad (1)$$

If the foreground-value is equal or higher than the background value, i.e. lighter foreground than background, the threshold function used is now the inverted binary operation (2), similar to the one described earlier.

$$bin(x,y) = \begin{cases} black & \text{if } src(x,y) > \text{foreground-value} \\ white & \text{otherwise} \end{cases} \quad (2)$$

While applying the FBCI method to each contour, in a rectangular region, an image is constructed with each individually binarized characters. Once this is done we have a rectangular region that consists only of black text over white background. This binary region is then forwarded to the *Tesseract* library, with the language set to Portuguese, in order to extract the included text. This extracted text is stripped of any leading and trailing whitespace and is appended to the rest of the newspaper content.

Since the rectangular regions do not contain all of the newspaper content, they are subtracted from the full front page image and the result is then passed through the *Tesseract* library and its text content is also appended to the newspaper content. Figure 6 displays the remaining news items on the front page, i.e. the subtraction of the front page with the detected rectangular regions, and a composite with this remaining image and the binarized items that were obtained earlier.

(a) Remaining news items on the front page.

(b) Binary rectangular regions added to the remaining items.

Fig. 6. Remaining news items and resulting final output.

4 Gazeta

The prototype newspaper archive was built using Ruby on Rails [7], which provides a stable starting point for development. Apache Solr [10] was used to store the information, obtained from the collection extractor. This collection extractor script iterates through all the collection images and feeds them to the newspaper-extractor application in order to obtain its content. This script then generates an XML file with the documents, which are then inserted into the Solr index. The application can then make requests to the Solr index, containing the keywords that the user wants to search for. The result from the requests is then parsed on a Ruby model and adjusted to be displayed on the view.

The views were built using Backbone.js [9], a JavaScript library that was used to enforce a coherent model structure while creating the single page application. Although this is a relatively recent, and not completely mainstream technology, it was used to create a more interactive interface, so that browsing the archive would still be an interesting activity. The occurrences graph was created with D^3 [4], a powerful and expansive JavaScript library that is used to visually display information.

The complete interface is shown in Fig. 7, where the user searched for the keyword *saúde*. Looking at the resulting graph it is clear that there is a peak in keyword occurrences around April, 2012.

Fig. 7. Occurrences for *saúde*.

5 Results

The annotated sample used for testing consists of 35 front pages, 5 from each of the daily newspaper publications: *Jornal de Notícias*, *Correio da Manhã*, *Público*, *Diário de Notícias*, *Jornal i*, *Destak* and *Metro - Grande Lisboa*. For each image, an XML file was created with all of the visible text in the front page, encompassing the content of the news items, advertisements and miscellaneous text, e.g. names of directors and editors, publication name, weekday of the issue.

The *Ruby* script iterates through each of the image files in the sample directory and feeds each one into the newspaper-extractor multiple times to test the different variations of the method and also the default test case. The script iterates through each of the words from the method's detected content file and checks if it is present on the sample list of words, if it is that word is removed from the sample and a correct match is counted. The accuracy value is the number of correct matches divided by the number of words on the sample.

Once all of the annotated sample images are processed the script then generates a CSV file with the mean accuracy value per publication, for each of the test cases. This CSV file can then be passed to an *R* [19] script which generates graphical visualisations of the results for quick visual comparison. Figure 8 and Fig. 9 display this graphical visualisation. The method variants present are:

FBCI: This is the default newspaper-extractor, with FBCI binarization only on the rectangular regions.

FBCI on rem.: This is similar to the previous variation but also applies the FBCI binarization method on the remaining content after the rectangular regions removal.

Global: This uses a global binarization method. Does not binarize the remaining image.

Tesseract: This is the default test case. Applying only the *Tesseract* library on the input image.

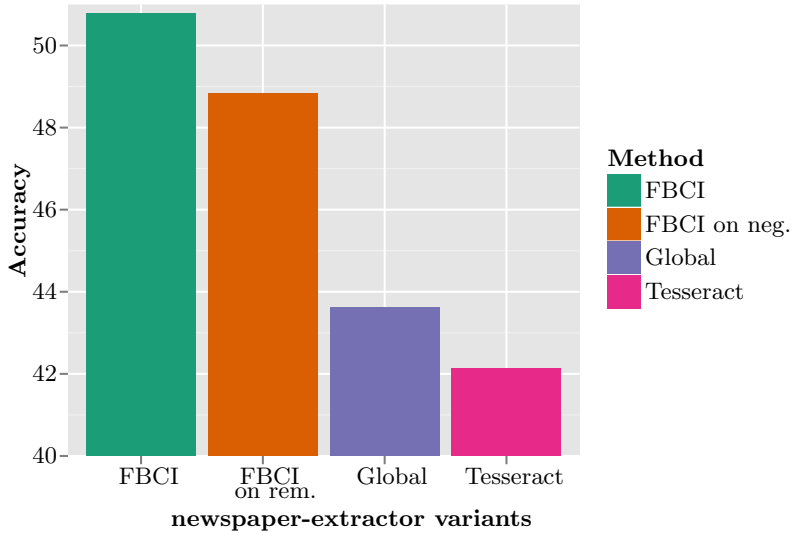


Fig. 8. Comparison between method variants.

The *Tesseract* test case, i.e. using only *Tesseract* to extract the keywords, resulted in an average accuracy of 42.12%. While using the newspaper-extractor resulted in an average accuracy of 50.74%, an increase of 8.62 percentage points.

The statistical significance of this improvement was verified using *R*'s one-sided paired t-test [15], to check if the mean accuracy of the newspaper-extractor sample results is greater than that of the *Tesseract* sample results. The resulting p-value is 8.678×10^{-06}, which supports the observation that the mean accuracy of the proposed keyword extraction method is greater than the mean accuracy of the *Tesseract* method with a confidence level of 99%.

6 Conclusion

Most of the reviewed newspaper archives have limited information, and limited search capabilities. This detracts users from using them for more than simple browsing of the daily headlines. Most segmentation methods used to extract information from newspapers and document images focuses on older newspapers and technical documents, dealing with stricter and simpler layouts without many images and colourful regions.

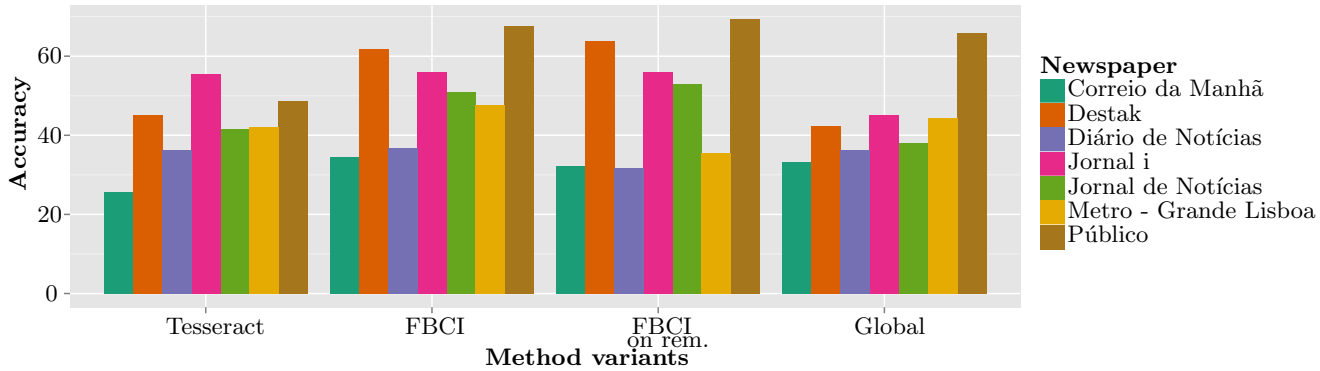


Fig. 9. Method variants impact on the major publications.

Using the newspaper-extractor, the keyword detection accuracy was improved by 8.62 percent points over using only *Tesseract* to extract the information. This improvement was verified as having statistical significance using a one-sided paired t-test. Developing a test bed from an annotated sample, despite being a tedious task, was very useful to keep track of the slight improvements on the application, and also to validate the benefits of a binarization method over another. The development from scratch of the *Gazeta* content-aware newspaper archive was harder than expected and the implemented features were also limited. However, it is still possible to discover interesting news item, since it is quick and easy to search for new content and to browse the results.

References

1. A. Antonacopoulos, B. Gatos, and D. Bridson. ICDAR 2007 Page Segmentation Competition. In *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, volume 2, pages 1279 –1283, September 2007. doi: 10.1109/ICDAR.2007.4377121.
2. Apostolos Antonacopoulos, Stefan Pletschacher, David Bridson, and Christos Papadopoulos. ICDAR 2009 Page Segmentation Competition. *2009 10th International Conference on Document Analysis and Recognition*, pages 1370–1374, 2009. doi: 10.1109/ICDAR.2009.275.
3. Fundaç ao Mário Soares. Diário de Lisboa. URL http://www.fmsoares.pt/diario_de_lisboa/ano. Accessed January 30, 2013.
4. Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. D^3: Data-Driven Documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–9, December 2011. ISSN 1941-0506. doi: 10.1109/TVCG.2011.185.
5. Krishnendu Chaudhury, Ankur Jain, Sriram Thirthala, Vivek Sahasranaman, Shobhit Saxena, and Selvam Mahalingam. Google Newspaper Search – Image Processing and Analysis Pipeline. In *2009 10th International Conference on Document Analysis and Recognition*, pages 621–625. IEEE, 2009. ISBN 978-1-4244-4500-4. doi: 10.1109/ICDAR.2009.272.

6. S. Chowdhury, S. Mandal, A. Das, and B. Chanda. Segmentation of Text and Graphics from Document Images. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007) Vol 2*, pages 619–623. IEEE, September 2007. ISBN 0-7695-2822-8. doi: 10.1109/ICDAR.2007.4376989.
7. Rails core team. Ruby on Rails. URL http://rubyonrails.org/. Accessed March 5, 2013.
8. Arganil City Counsil. A Comarca de Arganil. URL http://acomarcadearganil.cm-arganil.pt/. Accessed January 30, 2013.
9. DocumentCloud. Backbone.js. URL http://backbonejs.org/. Accessed March 5, 2013.
10. Apache Software Foundation. Apache Lucene - Apache Solr. URL http://lucene.apache.org/solr/. Accessed January 27, 2013.
11. gflags Dev Team. gflags - Commandline flags module for C++ - Google Project Hosting. URL https://code.google.com/p/gflags/. Accessed June 5, 2013.
12. Google. Google News Archive Search. URL http://news.google.com/newspapers. Accessed January 30, 2013.
13. T Kasar, J Kumar, and AG Ramakrishnan. Font and Background Color Independent Text Binarization. *Second international workshop on camera-based document analysis and recognition*, pages 3–9, 2007. URL http://www.imlab.jp/cbdar2007/proceedings/papers/O1-1.pdf.
14. Kiosko. Today's newspapers. Today's press covers. Kiosko.net, 2008. URL http://en.kiosko.net. Accessed January 30, 2013.
15. Todos Logos. Paired Student's t-test, July 2009. URL http://www.r-bloggers.com/paired-students-t-test/. Accessed June 5, 2013.
16. Newseum. Newseum — Today's Front Pages — Gallery View, 2001. URL http://www.newseum.org/todaysfrontpages/. Accessed January 30, 2013.
17. NewspaperArchive. Explore Historical Newspaper Archives Online. URL http://newspaperarchive.com/. Accessed January 30, 2013.
18. University of Salford. PRImA - Pattern Recognition & Image Analysis. URL http://www.primaresearch.org/. Accessed February 6, 2013.
19. R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2013. URL http://www.R-project.org. Accessed June 5, 2013.
20. SAPO. Nacional - Banca de jornais - SAPO Notícias. URL http://noticias.sapo.pt/banca/. Accessed February 5, 2013.
21. Stamen. In The News / Vox Delicii. URL http://stamen.com/projects/inthenews. Accessed June 11, 2013.
22. OpenCV Dev Team. OpenCV, August 2012. URL http://opencv.org/. Accessed February 4, 2013.
23. Tesseract Dev Team. tesseract-ocr - An OCR Engine that was developed at HP Labs between 1985 and 1995... and now at Google. - Google Project Hosting. URL https://code.google.com/p/tesseract-ocr/. Accessed February 4, 2013.
24. Marcos Weskamp. Newsmap, March 2004. URL http://marumushi.com/projects/newsmap. Accessed February 1, 2013.
25. InfoVis Wiki. "In The News" - Interactive Visualization of Google News, June 2005. URL http://www.infovis-wiki.net/index.php?title=2005-07-21:_"In_The_News"_-_Interactive_Visualization_of_Google_News. Accessed June 11, 2013.
26. Kerry Wong. Canny Edge Detection Auto Thresholding, May 2009. URL http://www.kerrywong.com/2009/05/07/canny-edge-detection-auto-thresholding/. Accessed March 20, 2013.