Automatic Extraction of Data Points and Text Blocks from 2-Dimensional Plots in Digital Documents

Saurabh Kataria and **William Browuer** and **Prasenjit Mitra** and **C. Lee Giles**
Pennsylvania State University, University Park, PA 16802, USA

Abstract

Two dimensional plots (2-D) in digital documents on the web are an important source of information that is largely under-utilized. In this paper, we outline how data and text can be extracted automatically from these 2-D plots, thus eliminating a time consuming manual process. Our information extraction algorithm identifies the axes of the figures, extracts text blocks like axes-labels and legends and identifies data points in the figure. It also extracts the units appearing in the axes labels and segments the legends to identify the different lines in the legend, the different symbols and their associated text explanations. Our algorithm also performs the challenging task of separating out overlapping text and data points effectively. Our experiments indicate that these techniques are computationally efficient and provide acceptable accuracy.

Introduction

With the advent of the world-wide-web, we have easy access to millions of documents in digital formats. In a document, authors often present their most important results in the form of figures (in scientific articles, financial reports, etc.). Among the more popular types of figures used to report experimental results and other data is a two-dimensional plot (2-D plot). In a 2-D plot, the variation of a dependent variable is plotted against the variation of an independent variable (see Figure 1). Though a human being can visually interpret a 2-D plot quite easily, the data in the plot are not available for automated processing. Algorithms to extract data from the deep web and also from tables on web documents have been proposed (Abiteboul, Buneman, & Suciu 2000)(Liu *et al.* 2007), to the best of our knowledge, there has been no significant efforts on extracting data from figures in digital documents.

We present a suite of image analysis and machine learning algorithms that extract data and metadata related to it from the figures and its captions. The tool based upon these algorithms extracts data from the 2-D plots and stores them in databases, so that this important source of data on the web can be searched (and eventually queried) using search engines. Specifically, the tool extracts the X and Y axis lines, ranges of values in the X and Y axes, the labels, units, and ticks on the axes, data points and lines in the plot, legends and the different types of data mentioned in the legend.

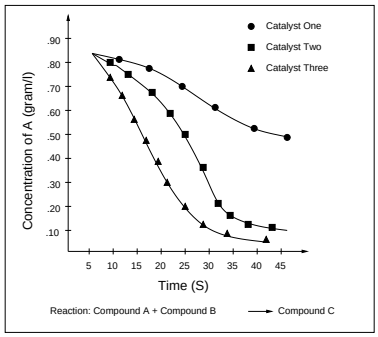


Figure 1: A sample 2-D plot describing the rate of a particular reaction in certain specific conditions

Extracting information from 2-D plots is a hard problem. Segmenting the image corresponding to the figure, extracting the different pieces of data available from the image, correlating the legend with the data, determining the units automatically using OCR, separating overlapping data points and text automatically are hard problems that involve extraction and interpretation of the data available in the image. Current search engines do not index information contained in the plots and make searching for these figures implicitly difficult. Our 2-D plot identification and extraction tool paves the first step towards building a database of extracted data from web documents.

The text present in the figure, i.e., legend, axis labels in addition to the caption of the figure are most suitable candidates to be incorporated in the meta-data of the 2-D figure. Extraction of text from half-tone (binary) and grey-scale images is a well explored area in the context of newspaper and magazines articles (Yuan & Tan 2001; Wang & Srihari 1989). However, the distribution of text

in 2-D figures is not as dense compared to magazines and newspapers and also, it is very likely that the position of the data points and the text in the 2-D figure is hard to predict beforehand which implies that the distribution of the data points in the 2-D figure presents a significant disturbance when trying to extract text from figures and vice-versa. Therefore, a probabilistic treatment of connected components is presented to separate the strings of text from the data points.

Our work is a first step towards making the data in figures from web documents available for querying and we recognize that there exist several other steps towards making this data fully available to end-users. However, we believe that drawing attention to the problem and providing the first set of tools to perform the challenging task of data extraction from figures in web documents is a significant contribution in itself.

Related Work

Content based image search and retrieval of image objects has been extensively studied (Datta, Li, & Wang 2005) (Doermann 1998). Using image understanding techniques, features are extracted from image objects such as texture features or color features and a domain specific indexing scheme is applied to provide efficient search for related image objects. For instance, (Manjunath & Ma 1996) utilize texture features to index arbitrary textured images, and (Smith & Chang 1996) utilize the spatial layout of regions to index medical images and maps. We believe the text present in different labels (i.e. legend, axis labels, caption) are most descriptive of the figure in general. Previous work in locating text in images consists of different application-based approaches such as page segmentation (Jain & Yu 1998), address block location (Jain & Bhattacharjee 1992), form (Yu & Jain 1996), and, color image-processing (Fletcher & Kasturi 1988a). Text can be located in an image using two primary methods (Jain & Yu 1998). The first treats text as textured region into the image and applies well-known texture filters such as Gabor filtering (Manjunath & Ma 1996), Gaussian filtering (Wu, Manmatha, & Riseman 1997), spatial variance (Zhong, Karu, & Jain 1995), etc. The second method uses connected components analysis (Yu & Jain 1996), (Jain & Bhattacharjee 1992), (Jain & Yu 1998), (Tang, Lee, & Suen 1996) on localized text regions and is applicable to binary images, i.e. images having two pixel intensity levels. Generally, both methods are used in conjunction to first isolate the suspected text regions using texture analysis and then using the binarized connected component analysis. These methods are highly application-sensitive and image structure drawn from domain knowledge is generally applied. Since most 2-D plots are best characterized by binary images, therefore we use connected component analysis for information extraction purposes.

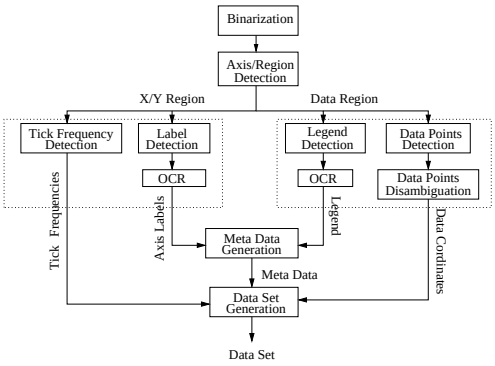


Figure 2: Process of Information Extraction from a 2-D plot

Problem Formulation

we define the problem and notations here that we use in the rest of this paper. A 2-D figure contains three regions 1) X-axis region containing X-axis labels and numerical units, i.e., area below the horizontal axis in Fig 1., 2) Y-axis containing labels and numerical units, i.e., the area to the left of vertical axis in Fig 1. and, 3) Curve region that contains legend text, and data points in case of curve-fitted plots or the curve itself, otherwise. A 2-D figure depicts a functional distribution of the form $y_i = f_i(x_j)\ with\ condition\ w_i$ where labels at the Y-axis and X-axis holds the text for y and x and legend text provides the particulars for conditions w. The values for these functions are represented by the data points or the curve in the plot. We use the term *plot metadata* to refer to these textual values that provide the semantic meaning to the data in the plot. In addition to the *plot metadata*, the *figure metadata* consists of the figure caption and the document level metadata for the figure.

Identification of a 2-D plot in digital document is a hard problem and is a pre-requisite to the information extraction problem. The identification problem is discussed in (Browuer *et al.* 2008). In this paper, we address two specific problems; 1) Metadata Extraction from the 2-D plot and its indexing, and 2) Data Extraction from the 2-D plot.

Information Extraction from 2-D Plots

Here we show how to automatically generate the metadata for the plot using the text that appears in the figure, and create a data set by identifying the data points. The metadata of a 2-D plot should capture parameters such as the dependent variable, the independent variable, conditions, etc.

The process flow of the information extraction is shown in the figure 2 and is explained below.

Plot Segmentation

Plot segmentation is the process of identifying and separating all the three regions, defined earlier, of a plot. Specifically, the algorithm must find the position of the coordinate axes in the plot. The coordinate axes act as a *global*

feature that identifies a 2-D plot. Profiling (or image signature) (Seul, O'Gorman, & Sammon 2000) is one important technique to detect global features if a rough measure of the image structure is known beforehand. This method is particularly suitable in our setting because almost all 2-D figures will have two perpendicular lines, the axes of the figures. The profile of a binarized image is calculated by counting all the foreground pixels along a specified image axis. In an image containing a single 2-D plot, the peak in the vertical profile corresponds to the Y-axis and the peak in the horizontal profile corresponds to the X-axis. The horizontal profile of the sample 2-D plot in fig. 1 is shown in the fig. 3(a). However, the profile based axis detection requires the plot axes to be aligned with image axes, but in the presence of the noise, e.g., in scanned images, the axes might not be perfectly perpendicular or aligned to the image axes, which may cause the proposed profiling technique to fail. Therefore, we need to perform a preprocessing step that enhances the global feature of the plot. This preprocessing step identifies the potential axes lines in the 2-D plot and reconstructs them aligned with the image axes. To determine axes lines, one needs to find the pair of set of maximum foreground pixels lying on straight line in the binarized image such that the slopes of two straight lines are almost orthogonal. This can easily be achieved by employing the Hough transformation to the original image (Duda & Hart 1972).

Hough transformation converts a binarized image from its x-y co-ordinate space to ρ-θ space (parametric space for straight line) where ρ,θ are the solution to the equation $xcos\theta + ysin\theta = \rho$. Thus, every point in ρ-θ corresponds to a set of collinear points in the x-y space. Also, the peaks in ρ-θ space corresponds to large segments of lines in x-y space. Since axis lines in a 2-D plot are one of the largest straight lines, therefore, the corresponding peaks are the brightest (or dominant) in ρ-θ space. Also, the brightest peaks closest to 0 and $\pi/2$ on $\theta-$axis will correspond to horizontal and verticle axis, respectively. The value of ρ for these peaks in ρ-θ space provides the perpendicular distance of the corresponding lines from origin in x-y space.

Furthermore, simple heuristics can be developed to segment figures with multiple 2-D graphs as there will be multiple global maximas in the profiles corresponding to the graphs. In this particular setting, an image is segmented into its constituent 2-D plots and each plot is further segmented into the three specified regions indicated above, which are separately analyzed in later stages.

Tick Detection Typically, tick marks on the axes denote intervals of measurement and usually appear with certain periodicity. We refer to this periodicity as tick frequency. If we can identify the coordinate location of the ticks on the axis, the values of the data points detected can be converted into their real values. The tick marks on the axes can be identified by performing Fast Fourier transforms on the axes profiles. Profile calculation has been described earlier. Once the axes have been identified and, thus, the coordinate origin, by considering the profile in the regions immediately surrounding x and y axes, the tick period or frequency can be determined. These profiles have a pulse train shape, where peaks correspond to major ticks. To ameliorate the effects of noise, and exploiting the simple relationship between a pulse train and its Fourier transform, the transform of the pixel projections are taken twice. The difference between successive peaks thus provides the numerical spacing of the major axis ticks. For tick frequency identification, we only consider the small width around the coordinate axes(we used 5 pixels in both directions) while calculating the axes profiles. The pseudocode for our algorithm is given below.

Algorithm *Axis Tick Frequency Identification*
Input: Binarized image segment S with dimensions ($\delta * \sigma$)
Output: Axis Tick Period and Pattern
1. Calculate axis profile $profileX[1\,..\,\sigma]$ = $Sum(S, \delta)$
2. Perform FFT twice, $pattern \leftarrow$FFT(FFT($profileX$))
3. invert sort; difference between index i and index j gives numerical period of axis ticks $[\beta_i, \beta_j, ..\, \beta_k]$ = inv(sort ($pattern$))
4. return $pattern(\beta_i\,..\,\beta_j)$

Text Block Extraction

Extracting text blocks from figures is more challenging than the extraction of text blocks from the raster image of a document, because in 2-D figures a) the distribution of text is sparse, and b) data points and lines introduce noise especially for the legend detection scheme. Note that a legend is usually a mixed block containing symbols representing different data types as well as text and has to be extracted as a single block. Therefore, the usual profile-based text-block detection techniques employed for extraction of text from documents (Jain & Bhattacharjee 1992) (Jain & Yu 1998) cannot be used. Because data and text both occur in the 2-D figure, it can not be directly processed by a standard optical character recognition (OCR) system. Considering these factors, we need to perform a connected component analysis to identify individual letters as a preprocessing step to recognize the text blocks that can, then, be sent to a standard OCR tool.

Usually, the text present in a 2-D figure does not contain any texture, i.e. significant intensity variation within the body of the text. Therefore, the loss of information is almost negligible if the image pixel intensities are converted into binary values. To perform component analysis, we apply the connected-component labeling scheme (Fletcher & Kasturi 1988b) that labels all the different connected components in the image. Text possesses certain spatial properties such as horizontal alignment and a certain spacing between characters, which distinguish the characters of a text string from other components in the image. These spatial characteristics of the text components can be utilized to perform the component analysis that provide the probabilistic location of the text in the 2-D plot. Specifically, we employ fuzzy rules based upon the spatial alignment of the characters to cluster the components into potential strings of text and then

employ OCR. The details are presented as follows.

Let $I(j,k)$ denote the raster matrix of the 2-D figure. We have observed that figures do not have text blocks that occupy very large areas in the figures. Therefore, after the connected component labeling, components that have an area above a certain threshold are not considered as candidates for being extracted as text blocks. The result of performing the connected component labeling is a numbered set of pixel matrices corresponding to each component, i.e. $CCL(I(j,k)) = \{C_i\}_{i=1}^{N}$, where the area of the components are below the threshold.

After obtaining the connected components representing single letters, we need to determine which letters are adjacent and part of the same text block and combine the components to create the block. We could determine some thresholds by which we would determine if two letters are adjacent. However, defining a fixed threshold would not lead to good accuracy because different figures use different fonts and spacing. Therefore, we want to use the distribution of the spacing and the vertical distance of the connected components to identify which letters are adjacent and which are parts of separate blocks. Let $X_i^h, X_i^l, Y_i^h, Y_i^l$ represent the corresponding maximum and minimum x and y coordinates of the i^{th} component, which indicate the height and width of a component. Let C_{ij}^y denote the vertical distance between any two components C_i and C_j i.e. $C_{ij}^y = Y_i^l - Y_j^l$; and C_{ij}^x denote the horizontal distance or *spacing* between C_i and C_j i.e. $C_{ij}^x = X_i^l - X_j^h$. The spatial alignment of the text characters can be encoded using the following fuzzy rule.

$$If\ C_{ij}^y \approx a\ And\ C_{ij}^x \approx b\ , Then\ C_i\ \leftarrow\ C_i \cup C_j \qquad (1)$$

where *a* and *b* approximate the spatial alignment parameters. The parameter *a* is the mean of the distribution of C_{ij}^y for all pairs of i and j. The parameter *b* is defined similarly. By applying the fuzzy rule, adjacent letters can be merged to create a text block. In some cases, the text appears vertically aligned, e.g., labels on the y-axis. For these characters, a similar fuzzy rule can be defined by replacing x with y in above equations. The fuzzy rule in Eq.1 can be encoded as a bivariate Gaussian unit which is

$$P(R_{ij}) = e^{-\frac{(C_{ij}^y - a)^2}{2s_1^2}} \cdot e^{-\frac{(C_{ij}^x - b)^2}{2s_2^2}} \qquad (2)$$

, where s_1 is the standard deviation corresponding to *a* and denotes the allowed difference, i.e, vertical spread, and s_2 is the standard deviation corresponding to *b* and denotes the allowed difference in x-dimension. R_{ij} is a random variable that gives the probability of C_i and C_j belonging to the same string or text block. However, the location of the strings of text in X and Y region is easy to predict using the region profile, for example, see the horizontal profile of only the X-axis region in Fig 3 b). This extra information is useful to accommodate the differences of various font shapes and can be used to approximate the values of parameters of Eq.2 on a per figure basis. Next, we show the algorithm for text detection in a 2-D plot.

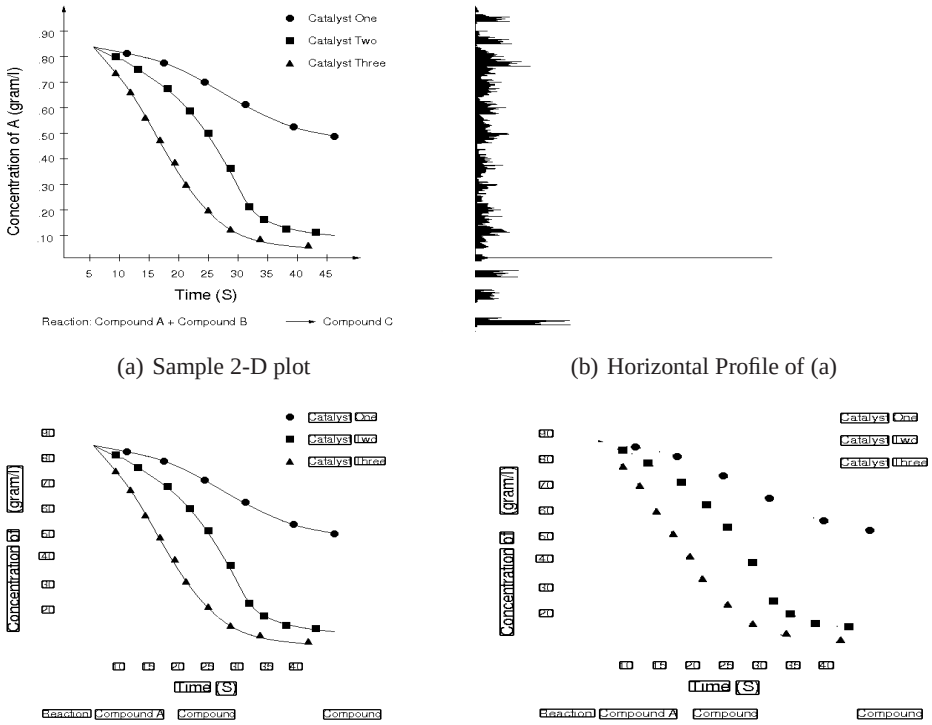
Algorithm *Text Detection in a 2-D Plot*
Input: Binarized Image matrix Segmented with Regions
Output: Strings set for OCR input.

1. **Initialize:** strings-set $\leftarrow\phi$; cc-set $\leftarrow\phi$; $\delta \leftarrow$20% of the image area
2. **For** X/Y Region
3. cc-set $\leftarrow$Connected Component Labeling
4. Detect the horizontal position of the text using horizontal profile for X region and Verticle profile for Y region.
5. separate the text strings using Verticle Profile of the text identified in previous step.
6. Approximate the parameters of Eq.2
7. **For** Curve Region
8. cc-set $\leftarrow$Connected Component Labeling
9. **for** each component C_i
10. **do if** Area$(C_i) > \delta$
11. **then** cc-set $\leftarrow$cc-set - C_i
12. **for** each pair of components C_i&C_j
13. **do** Calculate $P(R_{ij})$ using Eq(2).
14. **if** $P(R_{ij}) > 0.05$
15. **then** string-set $\leftarrow C_i, C_j$
16. strings-set $\leftarrow$string-set
17. **return** strings-set

Data Points Detection

Scatter and curve-fitted plots contain geometrical shapes that serve as data points in 2-D plots. Our next step after text block detection is to locate these data points. Isolation of these data points from the curves is essential to perform the shape detection in order to have a mapping with the legend. A reasonable heuristic that the curves have similar pixel width to the width of axes can be utilized to filter the lines in the curve region. We extend the basic idea of the k-median filtering algorithm (Seul, O'Gorman, & Sammon 2000) to perform this operation. To summarize the K-median algorithm, it is a first order pixel-noise filtering technique that removes isolated foreground pixels. A raster scan of the image is performed with a square window of $(k * k)$ size with $k = 2 * w + 1$ where w is set depending upon the noise in the image, e.g. w is set to 1 for 1-pixel noise. With each new position in the window, the intensity of the most central pixel in the window is assigned with the median intensity value of all the pixels that lie in that window.

The K-median filtering algorithm uses a 2-dimensional window, which is not able to preserve the contours of 2-dimensional shapes because pixels at edges of the shape are surrounded by a majority of background pixels. However, this problem can be overcome by taking a 1-dimensional filter and treating line width as the noise intensity. Therefore, we choose two windows of size $(1 * k)$ and $(k * 1)$ where $k = 2 * w + 1$ and then perform the raster scan of the image. Using two 1-dimensional windows preserves even the pixels at the edges of the two-dimensional data points reasonably well, but removes narrow lines from the figure (as desired). As pixel width and orientation of the data points has to be different from the curve and the average pixel width of the curve is almost similar to the axes width, therefore,w is set to be at most the value of axis width. We calculate the pixel width of the axes during the plot segmentation stage using



(a) Sample 2-D plot

(b) Horizontal Profile of (a)

(c) Text Detection

(d) Data Points Detection

Figure 3: Information Extraction process on a sample 2-D Figure

the profile of the 2-D plot. Figure 3(d) shows the result of the operation of two modified k-median filters. te Overlapping data points are common in 2-D plots and recognizing all the data points is especially challenging in this case. We employed unsupervised learning technique based upon simulated annealing to accurately determine the ground state and position of a geometric pattern (analogous to identifying the ground state of any physical system) of overlapping points. Interested readers should refer to (Browuer *et al.* 2008) for details of data point disambiguation algorithm.

Experiments

The data set for our experiments is randomly selected publications crawled from the web site of Royal Society of Chemistry(www.rsc.org) and randomly selected computer science publications from CiteSeer, digital library for scientific publications. Using (Browuer *et al.* 2008) classification scheme for detecting 2-D plots which has 88.25% recall, we collected 504 2-D plots.

Text Block Detection

Fig 3(c) shows the result of our text block detection algorithm on a sample 2-D plot shown in fig 3(a). We sampled 504 2-D figures for the text block detection. The algorithm segments each plot into three regions. Next, the algorithm observes the distribution of the vertical distance and the spacing of the letter components into X and Y axis and approximate mean and standard deviations. However, in almost 10% of the our figures, the location of the labels in X or Y region was hard to decide based upon the profiling scheme discussed earlier. This was because there were no global minima in the profile of the X or Y region, i.e., there were text positioned in the region in such a way that their profile overlapped. In such cases, the algorithm failed to identify the location of the labels in X & Y region and a pre-specified set of distribution parameters were assumed.

	Total	# Correct	% Recall
X Labels	504	428	84.9
Y Labels	504	441	87.5
Legend Text	504	398	79.0

Table 1: Experimental results of Text Block Detection Algorithm

We show the match results for the sample of 504 2-D plots in table 1. We consider the X or Y label or the legend text block correctly identified if, on an average, at least 70 % of the identified letters in the blocks are correctly extracted.

As we used the spatial alignment parameters for the text

Total	# Correct	% Recall
504	463	92

Table 2: Experimental results of Data Extraction Algorithm

block detection, some data points were incorrectly labeled as text also. Particularly, in 20 figures, that had very dense data point distribution, data points were incorrectly labeled as text blocks. However, data points can be filtered out, while we perform the OCR on the labeled text blocks.

There are many factors that affect the correctness of the text block detection algorithm. Presence of noise, data point interferences with the spatial alignment of the letters, different vertical alignment for certain letters e.g. *'y','j'* etc. are some of the prevalent factors. Although, we noticed a slight improvement in the accuracy measures with relaxation of the distribution parameter, however that improvement proved to be a trade-off in terms of labeling data points as text blocks. The complexity of the text block detection algorithm is $O(n)$, where n is the size of the image in pixels, on a single processor and $O(n * (1 + logp)/p)$, where p is the number of processors, on multiple processors. In case of multiple processors, the extra overhead of $(1 + logp)$ is due to the recursive solution for connected component labelling using divide and conquer strategy(Chaudhary & Aggarwal 1991).

Data Extraction

After filtering out the text blocks using the text block detection algorithm, we performed our data extraction algorithm on rest of the 2-D figure. Fig subfig:4 shows the result of the both text detection and data point detection algorithm. We used the 504 2-D figures which had scattered or curve-fitted plot in it. We call a figure an accept if more than 90 % of the data points gets correctly extracted with their shape preserved. Table 2 shows the recall for the data extraction algorithm. The complexity of the data point detection algorithm is $O(n)$, where n is the size of the image in pixels, on a single processor and $O(n/p)$, where p is the number of processors, on multiple processors.

Conclusion

We developed automated methods for extracting data and text from two-dimensional plots from digital documents and apply it to documents published on the web. This method eliminates the time consuming manual process of retrieving this data. The algorithm extracts axes, the ticks on the axes, and the text labels associated with the ticks as well as the labels of the axes. It identifies the legend as a text-dominated box in the figure and extracts the lines from the legend and segments the lines to extract each data point symbol and its textual description from the legend. Most importantly, our tool extracts data points from the plots, identifies their shapes and records the values of the X and Y coordinates for that point. We outline how the challenging problem of segmenting overlapping data points can be addressed. Experimental results show that the data and text extraction from the 2-D plots are fairly accurate. Future directions would include automatically extracting information from other graph figures.

Acknowledgments

This work supported in part by the National Science Foundation.

References

Abiteboul, S.; Buneman, P.; and Suciu, D. 2000. *Data on the Web: from relations to semistructured data and XML*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Browuer, W.; Kataria, S.; Das, S.; Mitra, P.; and Giles, C. L. 2008. Segregation and extraction of overlapping data points in digital documents. In *JCDL '08: Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries*. Pittsburgh, PA, USA: ACM.

Chaudhary, V., and Aggarwal, J. K. 1991. On the complexity of parallel image component labeling. In *ICPP (3)*, 183–187.

Datta, R.; Li, J.; and Wang, J. Z. 2005. Content-based image retrieval: approaches and trends of the new age. In *Multimedia Information Retrieval*, 253–262.

Doermann, D. S. 1998. The indexing and retrieval of document images: A survey. *Computer Vision and Image Understanding* 70(3):287–298.

Duda, R. O., and Hart, P. E. 1972. Use of the hough transformation to detect lines and curves in pictures. *Commun. ACM* 15(1):11–15.

Fletcher, L. A., and Kasturi, R. 1988a. A robust algorithm for text string separation from mixed text/graphics images. *IEEE Trans. Pattern Anal. Mach. Intell.* 10(6):910–918.

Fletcher, L. A., and Kasturi, R. 1988b. A robust algorithm for text string separation from mixed text/graphics images. *IEEE Trans. Pattern Anal. Mach. Intell.* 10(6):910–918.

Jain, A. K., and Bhattacharjee, S. K. 1992. Address block location on envelopes using gabor filters. *Pattern Recognition* 25(12):1459–1477.

Jain, A. K., and Yu, B. 1998. Document representation and its application to page decomposition. *IEEE Trans. Pattern Anal. Mach. Intell.* 20(3):294–308.

Liu, Y.; Bai, K.; Mitra, P.; and Giles, C. L. 2007. Tablerank: A ranking algorithm for table search and retrieval. In *AAAI-2007*, 317–322.

Manjunath, B. S., and Ma, W.-Y. 1996. Texture features for browsing and retrieval of image data. *IEEE Trans. Pattern Anal. Mach. Intell.* 18(8):837–842.

Seul, M.; O'Gorman, L.; and Sammon, M. J. 2000. *Practical algorithms for image analysis: description, examples, and code*. New York, NY, USA: Cambridge University Press.

Smith, J. R., and Chang, S.-F. 1996. Visualseek: A fully automated content-based image query system. In *ACM Multimedia*, 87–98.

Tang, Y.; Lee, S.; and Suen, C. 1996. Automatic document processing - a survey. In *Pattern Recognition*, volume 29-12, 1931–1952.

Wang, D., and Srihari, S. N. 1989. Classification of newspaper image blocks using texture analysis. *Computer Vision, Graphics, and Image Processing* 47(3):327–352.

Wu, V.; Manmatha, R.; and Riseman, E. M. 1997. Finding text in images. In *ACM DL*, 3–12.

Yu, B., and Jain, A. K. 1996. A generic system for form dropout. *IEEE Trans. Pattern Anal. Mach. Intell.* 18(11):1127–1134.

Yuan, Q., and Tan, C. 2001. Text extraction from gray scale document images using edge information. In *Sixth International Conference on Document Analysis and Recognition*, 302–306.

Zhong, Y.; Karu, K.; and Jain, A. K. 1995. Locating text in complex color images. In *ICDAR*, 146–149.