ICDAR 2003 Robust Reading Competitions

S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong and R. Young
Dept. of Computer Science
University of Essex, Colchester CO4 3SQ, UK

Abstract

This paper describes the robust reading competitions for ICDAR 2003. With the rapid growth in research over the last few years on recognizing text in natural scenes, there is an urgent need to establish some common benchmark datasets, and gain a clear understanding of the current state of the art. We use the term robust reading to refer to text images that are beyond the capabilities of current commercial OCR packages. We chose to break down the robust reading problem into three sub-problems, and run competitions for each stage, and also a competition for the best overall system. The sub-problems we chose were text locating, character recognition *and* word recognition.

By breaking down the problem in this way, we hope to gain a better understanding of the state of the art in each of the sub-problems. Furthermore, our methodology involves storing detailed results of applying each algorithm to each image in the data sets, allowing researchers to study in depth the strengths and weaknesses of each algorithm. The text locating contest was the only one to have any entries. We report the results of this contest, and show cases where the leading algorithms succeed and fail.

1. Introduction

Fifty years of research in machine reading systems has seen great progress, and commercial OCR packages now operate with high speed and accuracy on good quality documents. These systems are not robust, however, and do not work well on poor quality documents, or on camera-captured text in everyday scenes. The goal of general purpose reading systems with human-like speed and accuracy remains elusive. Applications include data archive conversion of noisy documents, textual search of image and video databases, aids for the visually impaired and reading systems for mobile robots.

In recent years there has been some significant research into these general reading systems that are able to locate and/or read text in scene images [11, 2, 3, 4, 1, 10]. So far, however, there have not been any standard publicly available ground-truthed datasets, which severely limits the conclusions which may be drawn regarding the relative merits of each approach.

Hence, the aims of these competitions are as follows:

- To capture and ground-truth a significant size text-in-scene dataset. This should have a shelf-life well beyond that of the competitions.
- To design or adopt standard formats for these datasets, and also for the results produced by the recognizers.
- To design or adopt standard evaluation procedures according to current best practices.
- To run the competitions in order to get a snapshot of the current state of the art in this area.

We aimed to broadly follow the principles and procedures used to run the Fingerprint Verification 2000 (and 2002) competitions [6]. Well in advance of the deadline we published sample datasets for each problem, the evaluation software to be used, and the criteria for deciding the winner of each contest. To enter the contests, researchers had to submit their software to us in the form of a ready-to-run command-line executable. This takes a test-data input file and produces a raw results file. The raw results are then compared to the ground truth for that dataset by an evaluation algorithm, which produces a set of detailed results and also a summary. The detailed results report how well the algorithm worked on each image, while the summary results report the aggregate over all the images in the dataset. All these files are based on simple XML formats to allow maximum compatibility between between different versions of evaluation systems, recognizers and file formats. In particular, new attributes and elements can be added to the markup while retaining backward compatibility with older recognition systems. The generic process is depicted in Figure 1.

2. Data Capture

Images were captured with a variety of digital cameras by each of the authors. Cameras were used with a range of resolution and other settings, with the particular settings chosen at the discretion of the photographer.

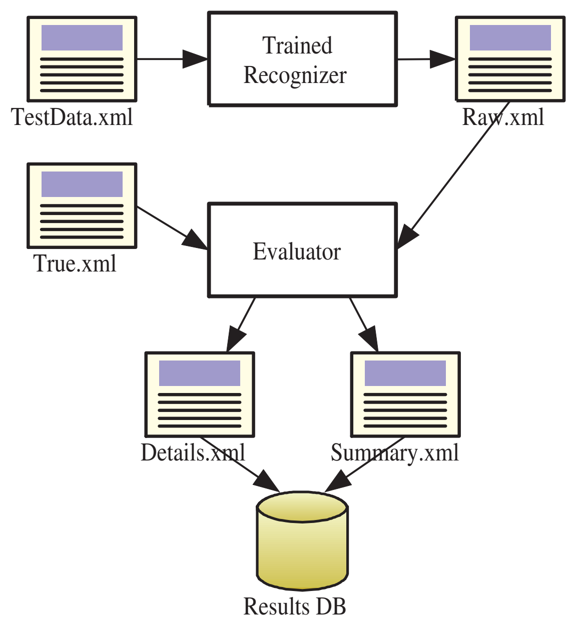

Figure 1. The multi-stage evaluation process.

To allow management of the ground-truthing or tagging of the images, and with an view to possible future tagging jobs, we implemented a web based tagging system. This operates along similar lines to the OpenMind[8] concept. Taggers can log in to the system from anywhere on the Internet using a Java (1.4) enabled web-browser. On logging in, a Java Applet window appears and presents a series of images. The tagger tags each image by dragging rectangles over words, and then typing in the associated text. The applet then suggests a possible segmentation of the word into its individual characters, which the tagger can then adjust on a character-by-character basis. The tagger can also adjust the slant and rotation of the region. When the tagger has finished an image, they click submit, at which point all the tagged rectangles are sent back to a server, where they are stored in a database. One of the parameters of the system is how many taggers should tag each image. If we had a plentiful supply of tagging effort, then we could send each image to several taggers, and simply accept all the images where the tags from different taggers were in broad agreement. This is somewhat wasteful of tagging effort, however, since it is much quicker to check an image than it is to tag it. We therefore adopted a two-tier tagging system of taggers and checkers, where the job of a checker was to approve a set of tags.

There are several ways of communicating between the applet and the server. We chose to use Simple Object Access Protocol (SOAP) - partly to gain experience of SOAP on a real project, and partly to allow good interoperability with other systems. Potentially, someone could now write a tagging application in some other language, and still request images to tag, and upload tagged images to our server.

3. The Competitions

Reading the text in an image is a complex problem that may be decomposed into several simpler ones. The best way to do this decomposition is open to debate. We chose to break down the robust reading problem into three stages, and run competitions for each stage, and also a competition for the best overall system. The stages we chose were text locating, character recognition and word recognition. Another possible stage would have been segmentation of words into separate characters. This idea was rejected on the grounds that we believed the images would be too difficult to segment in a way that was independent of the OCR process, and we also wanted to place some limit on the number of competitions to be run. However, the segmentation data exists for all the words in the database, so it is still possible for researchers to evaluate their segmentation algorithms on this data. For each of the competitions we describe the proposed performance measures and time limits. We also set up an on-line forum to allow competitors to discuss these, but there forum was hardly used, and most discussion was conducted via email.

3.1. Pre-trained Systems

For all the competitions we debated whether to run them for trainable or non-trainable systems. We decided that any system training or tuning was best left to the system designers, and hence each of the contests deals with evaluating pre-trained systems. The contestants were advised to download the trial datasets well in advance of the competition deadline, in order to tune their systems for optimal performance on this type of data.

From a machine learning perspective it would be desirable to test the learning ability of each method. Our prime concern here, however, is to find the system that performs best on each task, irrespective of the amount of hand-tuning that went into its design. Hence we justify our decision to base the contests on pre-trained systems.

3.2. Text Locating

The aim of the text locating competition is to find the system that can most accurately identify the word regions in an image.

For this contest, a text locating algorithm takes a JPEG file as input and produces a set of rectangles as output. The preferred system interface is that both the input and output files are in a simple XML format, described on the contest web page. Taking the example image in Figure 2, a text locating algorithm would ideally identify five rectangles in image pixel coordinates, surrounding the words: ”Department”, ”of”, ”Computer”, ”Science”, ”1”.

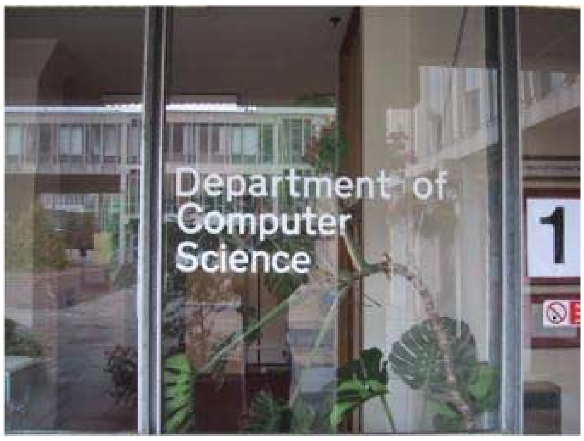

Figure 2. Example scene containing text.

Note that several design options were possible here - such as specifying that the system find complete text blocks, or individual words or characters. We chose words since they were easier to tag and describe (it would be harder to fit rectangles to text blocks, since they are more complex shapes).

We aimed to design an evaluation scheme that would be:

- Easy to understand and to compute;
- Reward text locating algorithms that would be most useful as a component of a text-in-scene word recognizer;
- Heavily punish any trivial solutions (e.g. such as returning a single rectangle covering the entire image, or returning all the possible rectangles for a given size of image).

The proposed evaluation system is based on the notions of precision and recall, as used by the information retrieval community. An alternative form of evaluation would be a goal-directed approach [9]. In this case, the text locating algorithms could be judged by the word recognition rate they achieve when used in conjunction with a word recognizer (or OCR package). A difficulty of this approach, however, is its dependence on the particular recognizer used. A detailed evaluation of various object detection evaluation methods is given in [7].

In general, precision and recall are used to measure a retrieval system as follows. For a given query (in this case, find all the word-region rectangles in an image), we have a ground-truth set of targets T and the set returned by the system under test, which we call estimates, E. The number of estimates which are correct, we denote c.

Precision, p is defined as the number of correct estimates divided by the total number of estimates:

$$p = \frac{c}{|E|}$$

Systems that over-estimate the number of rectangles are punished with a low precision score.

Recall, r is defined as the number of correct estimates divided by the total number of targets.

$$r = \frac{c}{|T|}$$

Systems that under-estimate the number of rectangles are punished with a low recall score.

For text locating it is unrealistic to expect a system to agree exactly with the bounding rectangle for a word identified by a human tagger. Hence, we need to adopt a flexible notion of a match. We define the match m_p between two rectangles as the area of intersection divided by the area of the minimum bounding box containing both rectangles. This figure has the value one for identical rectangles and zero for rectangles that have no intersection. For each rectangle in the set of estimates we find the closest match in the set of targets, and vice versa.

Hence, the best match $m(r, R)$ for a rectangle r in a set of Rectangles R is defined as:

$$m(r, R) = \text{max } m_p(r, r') \mid r' \in R$$

Then, our new more forgiving definitions of precision and recall:

$$p' = \frac{\Sigma_{r_e \in E}\, m(r_e, T)}{|E|}$$

$$r' = \frac{\Sigma_{r_t \in T}\, m(r_t, E)}{|T|}$$

We adopt the standard f measure to combine the precision and recall figures into a single measure of quality. The relative weights of these are controlled by α, which we set to 0.5 to give equal weight to precision and recall:

$$f = \frac{1}{\alpha/p' + (1-\alpha)/r'}$$

We had planned to impose a time-limit of 10s per image on average, but dropped this as some of the systems submitted were unable to comply with this, and given the small number of entries, we felt it would be inappropriate to be too strict.

3.3. Robust Reading

The aim of this competition is to find the best system able to read complete words in camera captured scenes.

Taking the example image in Figure 2, it would ideally identify five words: ”Department”, ”of”, ”Computer”, ”Science”, ”1”, and also specified a bounding rectangle (in image pixel coordinates) for each word.

Note that the Text Locating, Robust Character Recognition and Robust Word Recognition all tackle sub-parts of this problem. The robust reading competition aims to identify the system that best does the complete job. The robust reader takes as input a scene image, and produces a set of tagged rectangles as output, where each rectangle is tagged with a single word hypothesis. The standard measures of precision and recall are used to evaluate the performance of a Robust Reader. Unlike the text locating contest, where we rate the quality of match between a target and estimated rectangle, we define a strict notion of match between the target and estimated words: the rectangles must have a match score m_p (see below) of greater than 0.5, and the word text must match exactly. The winning system would be the one with the best f score.

3.4. Robust Word and Character Recognition

The aim of these competitions is to find the system best able to read single words that have been extracted camera-captured scenes. The word recognizer takes two inputs: a file of words to be recognized, and a dictionary file. For these experiments a custom dictionary is supplied that has 100% coverage of the words in the images. The term *word* is used loosely here to mean any string of characters that the image taggers approved as a word, and some of the character strings would not be in a conventional dictionary. To simplify our software, we designed the character recognizer interface to operate in an identical manner to the word recognizer, except that words are restricted to be one character in length. Despite several expressions of interest, we received no submissions for these contests in time to include in this paper. Example word and character images or shown in Figures 3 and 4 respectively.

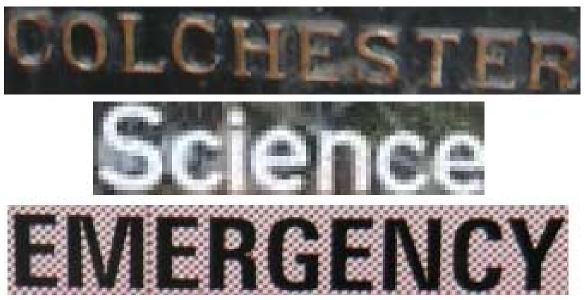

Figure 3. Example extracted words.

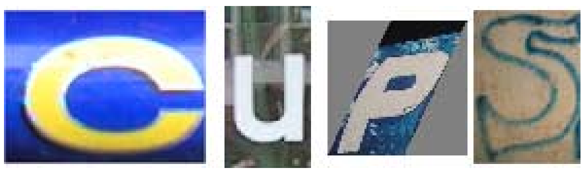

Figure 4. Example extracted characters.

Problem	Downloads	EOIs	Entries
locating	394	7	5
word	228	4	0
char	218	5	0

Table 1. Measures of interest in each problem.

4. Experimental Setup

We organized the data for each competition into: *Sample*, *Trial* and *Competition* datasets. Sample datasets were provided to give a quick impression of the data, and also to allow functional testing of your software i.e. researchers could check that their software could read and write the specified dataset formats, but not get any statistically meaningful results.

Trial datasets had two intended uses. They could be used to get results for ICDAR 2003 papers. For this purpose, they were partitioned into two sets: *TrialTrain* and *TrialTest*. The instructions were to use TrialTrain to train or tune algorithms, then quote results on TrialTest. For the competitions, the instructions were that algorithms should be trained or tuned on the entire trial set (i.e. TrialTest $\cup$ TrialTrain).

Competitors were then invited to submit their tuned systems by the competition deadline of April 30th, 2003. The submissions were then evaluated by running them on the competition datasets.

Table 1 gives an idea of the level of interest in each problem. The downloads column shows the number of downloads of the sample dataset for each problem; in each case, the number of downloads of the trial datasets, which are much larger, was approximately half this figure. Note that the text locating dataset (*locating*, in the table) was the same as the robust reading dataset, but there were no expressions of interest in the robust reading problem. Note that only in the case of the text locating problem did the expressions of interest (EOIs) translate to actual entries. For the other contests we are extending the deadline in the hope of attracting some entries.

5. Results

The Text Locating competition had five entries by the April 30th deadline; the other contests all had zero entries. Many of the originally supplied entries were missing DLL or other library files - contestants were invited to supply any missing files, which they all did. Some of the originally supplied systems were buggy, and would crash after processing several images, perhaps due to memory leaks. Again, contestants were invited to supply fixed versions, which they mostly did. In the case of one of the submissions, the patched version still crashed frequently, and had

System	precision	recall	f	t (s)
Ashida	0.55	0.46	0.50	8.7
HWDavid	0.44	0.46	0.45	0.3
Wolf	0.30	0.44	0.35	17.0
Todoran	0.19	0.18	0.18	0.3
Full	0.1	0.06	0.08	0.2

Table 2. Text Locating competition results.

such a low score ($f = 0.01$) on a set of sample images that we did not run it on the full set, and hence did not include it in the results table.

In future it would be much less effort if we could run these competitions by using an alternative mode of entry, where each competitor exposes their system as a web service [5], which they are responsible for maintaining. The evaluation program would then work by supplying images to the service, which aims to return the set of rectangles for each image in XML. We aim to foster this approach by supplying some skeleton software and examples of how to do this.

5.1. Text Locating Results

The Text Locating results on the competition data are shown in Table 2. The entries are identified by the user name of the person submitting each one. The entries all appear to be from academic institutions, with affiliations as follows: *Ashida*, Department of Computer Engineering, Shinshu University, Japan; *HWDavid*, Institute of Automation, Chinese Academy of Science; *Wolf*, Institut National de Sciences Appliquées de Lyon, France; *Todoran*, Department of Computer Science and Logic, University of Amsterdam, Netherlands.

The column labelled $t(s)$ gives the average time in seconds to process each image for each system under test. Note that the *Full* system is the score obtained by returning a single rectangle for each image that covers the entire image. This could have been computed from the *resolution* information in the XML input file, but to give a baseline measure of the time we computed this by retrieving and decompressing each JPEG image, then measuring the image size.

Note that poor performance under our evaluation scheme does not necessarily mean that the algorithms are poor at finding text. For example, in some of the results of Todoran that we studied, the algorithm had tagged a large block of text consisting of multiple words with a single rectangle. Our evaluator gives some credit for this, but not nearly as much as a locater that identifies individual words, which was the object of the test.

We viewed many of the results of each program, especially the two leaders, to gain an impression of the strengths and weaknesses of each system. In each of the following images the ground truth rectangles are identified with black dashed lines, while the estimated rectangles are shown with magenta (light grey) dashed lines.

Figure 5 shows the output of HWDavid on an image where both HWDavid and Ashida perform poorly. On this test HWDavid identifies lots of false text rectangles, while Ashida returns just one rectangle that has no intersection with the ground truth rectangle ("TAXI").

All the algorithms under test are somewhat inconsistent in their ability to locate text. In some cases they detect noisy, hard-to-read text, while in other cases they miss text that to the human eye is very clear. For example, HWDavid detects some of the text in Figure 6, while missing other parts such as *SV* that are in the same font and appear equally clear. HWDavid has an f-score of 0.65 for this image, while Ashida returns no rectangles and scores 0.0.

Figure 7 shows a case where Ashida correctly locates the text ("15") in the image and achieves an f-score of 0.88, but HWDavid returns no rectangles and scores 0.0.

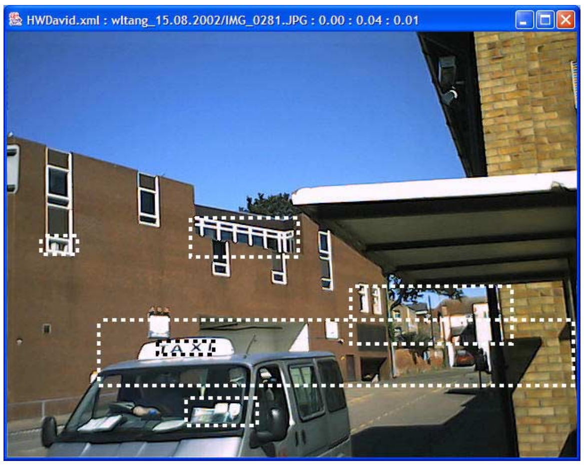
HWDavid.xml : wltang_15.08.2002/IMG_0281.JPG : 0.00 : 0.04 : 0.01
TAXI

Figure 5. An image on which both leading algorithms score poorly.

6. Discussion and Conclusions

Our main intention in running these competitions was to gain a clear picture of the state of the art of reading text in scenes. This has so far been partially successful for the text locating problem, but not for the other problems. The public datasets we captured and tagged, should make a useful contribution to research in this area.

Running the text locating contest has given us some tentative insights into the strengths and weaknesses of the submitted systems. These can be summarized as follows:

- Even the best performing systems are inconsistent, detecting some text while missing apparently very similar text in the same image.
- There was a major difference in the speed of the submitted systems, with Ashida being over twenty times

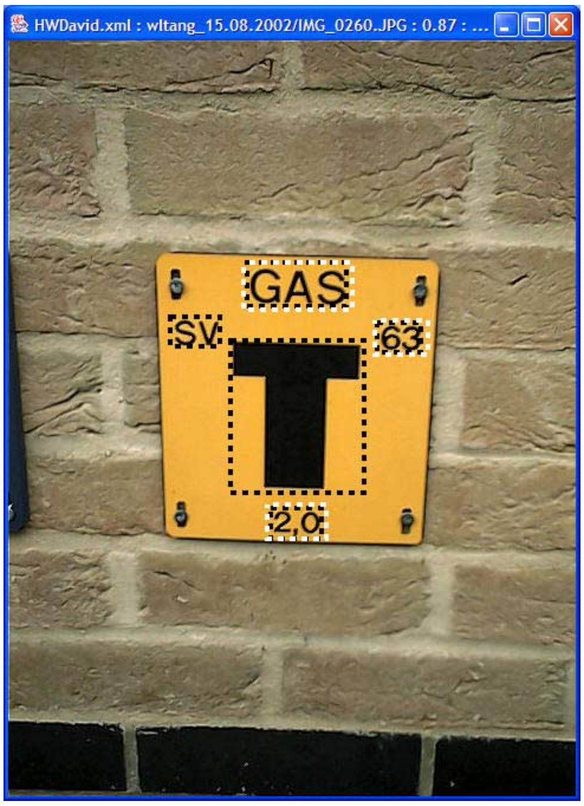

Figure 6. An image where HWDavid beats Ashida.

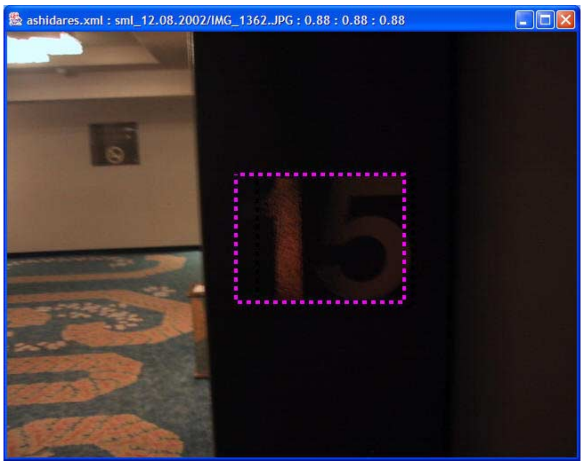
ashidares.xml : sml_12.08.2002/IMG_1362.JPG : 0.88 : 0.88 : 0.88
15

Figure 7. A hard to read image that Ashida does well on, while HWDavid scores zero.

slower than HWDavid, though a little more accurate.

- Variations in illumination, such as reflections from light sources cause significant problems.
- Variations in scale cause significant problems, in that the same image presented to a system at a different, but equally readable scale, causes different regions to be identified by the algorithms.

Reading text in scenes, and other noisy images, is still very much a challenging problem. We believe that the results of the text locating contest give an idea of the state of the art in this particular sub-problem.

Acknowledgements We are grateful to The Government Communications Headquarters (GCHQ), Cheltenham, UK for funding the software development and data capture for these competitions.

We thank the participants for their considerable effort in taking part.

References

[1] J. Chang, X. Chen, A. Hanneman, J. Yang, and A. Waibel. A robust approach for recognition of text embedded in natural scenes. *Proceedings of International Conference on Pattern Recognition*, 2002.

[2] P. Clark and M. Mirmehdi. Combining statistical measures to find image text regions. In *Proceedings of the 15th International Conference on Pattern Recognition*, pages 450–453. IEEE Computer Society, 2000.

[3] H. Li, D. Doermann, and O. Kia. Automatic text detection and tracking in digital videos. *IEEE Transactions on Image Processing*, 9(1):147–156, 2000.

[4] R. Lienhart and A. Wernicke. Localizing and segmenting text in images and videos. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(4):256 – 268, 2002.

[5] S. Lucas. Web-based evaluation and deployment of pattern recognizers. *Proceedings of International Conference on Pattern Recognition*, pages 419 –422, 2002.

[6] D. Maio, D. Maltoni, R. Cappelli, J. Wayman, and A. Jain. Fvc2000: Fingerprint verification competition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:402 – 412, (2002).

[7] V. Mariano, J. Min, J. Park, R. Kasturi, D. Mihalcik, H. Li, D. Doermann, and T. Drayer. Performance evaluation of object detection algorithms. In *Proceedings of the International Conference on Pattern Recognition*, pages 965–969, (2002).

[8] D. Stork. The open mind initiative. `http://www.openmind.org`.

[9] O. Trier and A. Jain. Goal-directed evaluation of binarization methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:1191–1201.

[10] C. Wolf, J.-M. Jolion, and F. Chassaing. Text localization, enhancement and binarization in multimedia documents. In *Proceedings of the International Conference on Pattern Recognition*, volume 4, pages 1037–1040, (2002).

[11] V. Wu, R. Manmatha, and E. M. Riseman. Finding text in images. In *Proceedings of 2nd ACM Conference on Digital Libraries*, pages 3–12, 1997.