A Fast Image-Gathering System on the World-Wide Web Using a PC Cluster

Keiji Yanai, Masaya Shindo and Kohei Noshita

Department of Computer Science, The University of Electro-Communications
1-5-1 Chofugaoka, Chofu-shi, Tokyo 182-8585, JAPAN
E-mail: {yanai,shindo-m,noshita}@igo.cs.uec.ac.jp

Abstract. Thanks to the recent explosive progress of WWW (World-Wide Web), we can easily access a large number of images from WWW. There are, however, no established methods to make use of WWW as a large image database. In this paper, we describe an automatic image-gathering system from WWW, in which we use both keywords and image features. By exploiting some existing keyword-based search engines and selecting images by their image features, our system obtains, with high accuracy, images that are strongly related to query keywords. This system has been implemented on a parallel PC cluster, which enables us to gather more than one hundred images from WWW in about one minute.

1 Introduction

Thanks to the recent explosive progress of WWW (World-Wide Web), we can easily access a large number of images from WWW. Hence, we can regard WWW as a huge image database. However, most of those images on WWW are not classified with appropriate keywords.

We can use commercial search engines for searching WWW for HTML documents by giving them related keywords. In a similar way, we can also use some image-search engines for searching WWW for images related to keywords. Most of image-search engines, however, search for images by using only keywords in HTML documents including images, without analyzing the contents of those images. As a result, they tend to return images that are not appropriate images to the given keywords.

As a method of image-searching, content-based image retrieval (CBIR) has been investigated [1, 2]. The conventional keyword-based image search methods require appropriate keywords, attached to all images in a database, which have to be made by hand in advance, whereas CBIR does not require such keywords. In CBIR, some types of similarity between images are computed using image features extracted from images. Thus, we can search for images similar to query images.

For constructing an image-search system on WWW based not only on keywords but also on the contents of images, in this paper, we propose an automatic image-gathering system on WWW, into which we have integrated a keyword-based search method and a CBIR method. In our system, a user gives query keywords to the system at the beginning of a search, and obtains output images related to the keywords. We have implemented the system on a PC cluster as

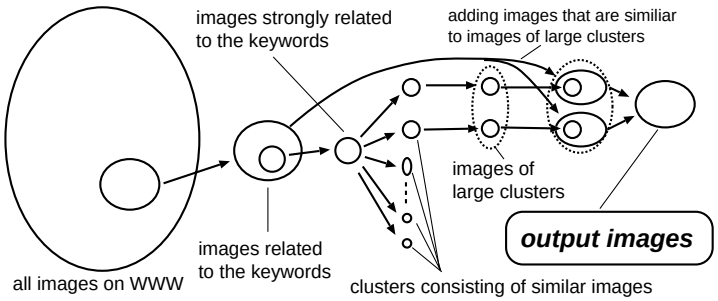

Fig. 1. Processing flow of image-gathering.

a parallel system for achieving fast image-gathering, which enables us to gather more than one hundred images from WWW in about one minute. In this paper, we describe our method of gathering images from WWW, an implementation of the system and results of experiments.

2 Our Method of Image-Gathering

The final goal of our image-gathering system is to gather images on WWW related to the query keywords given by a user as input. Note that our system is not called an image "search" system but an image "gathering" system, since our system has the following properties: (1) it does not search for images over the whole WWW directly, (2) it does not make a database in advance, and (3) it makes use of search results obtained by commercial keyword-based text-search engines.

Figure 1 shows the processing flow. Since an image on WWW is usually embedded in an HTML document that explains it, the system exploits some existing commercial keyword-based WWW search engines, and it gathers URLs (Universal Resource Locator) of HTML documents related to query keywords. In the next step, using those gathered URLs, the system fetches HTML documents from WWW, analyzes them, and evaluates the extent of relation between the keywords and images embedded in HTML documents. If it is judged that images are related to keywords, the image files are fetched from WWW. According to the extent of relation to the keywords, we divide fetched images into two groups: images in group A having stronger relation to the keywords, and others in group B. For all gathered images, image features are computed.

In CBIR, a user has to provide query images to the system, because it searches for images based on the similarity of image features between query images and images in an image database. In our system, instead of providing query images, a user only needs to provide query keywords to the system. Then, we select images strongly related to the keywords as group A images, remove noise images from them, and regard them as query images only by examining keywords. Removing noise images is carried out by eliminating images which belong to relatively small clusters in the result of image-feature-based clustering for group A images. Images which are not eliminated are regarded as appropriate images to the query

keywords, and we store them as output images. Our preference of larger clusters to smaller ones is based on the following heuristic observation: an image that has many similar images is usually more suitable to an image represented by keywords than one that has only a few similar images. Next, we select images that are similar to the query images from group B in the same way as CBIR, and add them to output images.

Some WWW image search systems such as WebSeer[3], WebSEEk[4] and Image Rover[5] have been reported so far, which can be regarded as an integration of keyword-based search and content-based image retrieval. These systems search for images based on the query keywords, and then a user selects query images from search results. After this selection by the user, the systems search for images that are similar to the query images based on image features. These three systems carry out their search in an interactive manner. Our system is different from those in that our system only needs one-time input of query keywords. Our system is able to gather a large number of various images related to the keywords, since it is unnecessary for a user to indicate query images during the processing, and the whole processing is executed automatically. The three systems quoted above require gathering images over WWW in advance and making large indices of images on WWW. In contrast to those systems, due to exploiting existing keyword-based search engines, our system does not require making a large index in advance.

3 Collection and Selection

The image-gathering process in our system consists of a collection part and a selection part.

3.1 Collection Part

In the collection part, by means of some commercial keyword-based WWW search engines, the system obtains URLs, and then, by using those URLs, it gathers images from WWW. The algorithm is as follows:

1. A user provides the system with query keywords.
2. The system sends queries to commercial keyword-based search engines, and obtains URLs of HTML documents related to the keywords.
3. The system fetches HTML documents indicated by the URLs from WWW.
4. The system analyzes HTML documents, and extracts URLs of images embedded in the HTML documents with image-embedding-tags ("`IMG SRC`" and "`A HREF`"). For each of those images, the system calculates a score which represents the intensity of relation between the image and the query keywords. The score is calculated by checking the following conditions:

   Condition 1: Each time one of the following conditions is satisfied, 3 points are added to the score.
   - In case the image is embedded by "`SRC IMG`" tag, "`ALT`" field of "`SRC IMG`" includes the keywords.
   - In case the image is linked by "`A HREF`" tag directly, words between "`A HREF`" and "`/A`" include the keywords.

– The name of the image file includes the keywords.

Condition 2: Each time one of the following conditions is satisfied, 1 point is added to the score.

– "`TITLE`" tag includes the keywords.
– "`H1, ..,H6`" tags include the keywords, if these tags are located just before the image-embedding-tag.
– "`TD`" tag including the image-embedding-tag includes the keywords.
– Ten words just before the image-embedding-tag or ten words after it include the keywords.

If the final score of an image is higher than 3, the image is classified into group A. If it is higher than 1, the image is classified into group B. The system fetches only image-files whose image belongs to either group A or B. If the size of a fetched image-file is larger than a certain predetermined size, the image is sent to the selection part.

5. In case the HTML document does not include image-embedding-tags at all, the system fetches and analyzes other HTML documents linked from it in the same manner described above, provided that it includes a link tag ("`A HREF`") which indicates URL of HTML documents on the same web site.

3.2 Selection Part

In the selection part, the system selects appropriate images for the query keywords out of images which are collected in the collection part. The selection is based on the image features as described below.

1. In the first step, for each of the collected images, the system makes a color histogram as image features [6]. Rather than making a color histogram directly for the RGB color space, we make it for the Lu^*v^* color space into which the RGB color space is converted. The reason for this is that the Lu^*v^* color space is known to represent the human color sense better than the RGB color space [7]. We quantize the Lu^*v^* color space into 216 (6 for each axis) bins, and make a color distribution histogram for each image. In the current implementation, we use these simple image features, although we can use other sophisticated image features proposed in many CBIR researches.
2. For each pair of images in group A, the distance which represents the degree of dissimilarity between the two images is calculated based on their image features. In the calculation of the distance, we do not adopt the Euclid distance but the distance which considers the proximity in the color space [8].
3. Based on the distance between images, images in group A are clustered by the cluster analysis method. Since we intend to make clusters so that images in the same one are similar to each other, we adopt the farthest neighbor method (FN): we define the distance between clusters as the largest distance between two images belonging to mutually different clusters. In the beginning, each cluster has only one image. For each pair of clusters, if the distance between them is smaller than a certain threshold, they are merged into the same cluster. The system repeats merging clusters, until all distances between clusters are more than the threshold.

4. The system throws away small clusters which have fewer images than a certain threshold value. It stores all images in the remaining clusters as output images.
5. The system selects images in group B if they have a small distance from images in the remaining clusters of group A, and adds them to output images.

4 Implementation

In our system, unlike the conventional image search systems for WWW, we do not make any index of images in advance, and we gather images from WWW on demand.

Because of this, the image-gathering process takes much longer time than that of the conventional systems. In order to speed up the whole process, we implement our system on a PC cluster, by which we achieve not only parallel processing within the collection part but also concurrent processing of the collection and selection parts.

Parallel processing within the collection part means that the system generates many collection processes on multiple PCs, and they gather images from WWW in parallel.

Concurrent processing of the collection and selection parts means that, before all the constituent processes of the collection part terminate, some processes in the selection part start. In the collection part, the process terminates when a collection process has collected all HTML documents and image files indicated by URLs. This implies that the load of the system gradually decreases, since the number of active collection processes decreases as time progresses. When the system starts access to HTML documents of all URLs obtained from text-based search engines, without waiting for the completion of fetching all the images, we can start to extract image features from images which have been already fetched and also compute distances between images in group A (Figure 2).

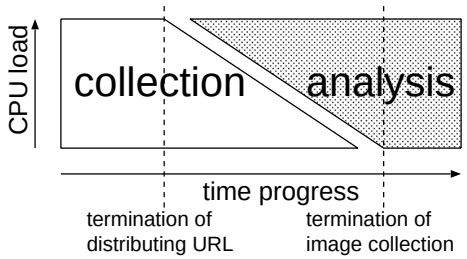

Fig. 2. Concurrent processing of the collection part and the analysis part.

The system consists of a master PC and some slave PCs as shown in Figure 3. The master PC issues search requests to keyword-based WWW search engines, manages URLs of HTML documents related to the keywords returned from the search engines, and select images from group A and B sent from slave PCs based on their images features.

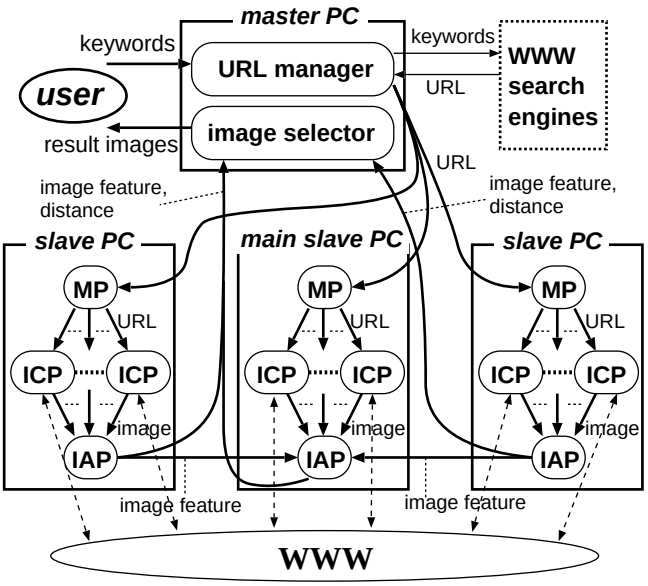

Fig. 3. Parallel image-gathering system.

Each slave PC has one management process(MP), some image collection processes(ICP) and one image analysis process(IAP).

An MP receives URLs from the master PC, and distributes them to ICPs in the same slave PC. Each ICP fetches HTML documents indicated by URLs handed by the MP, extracts URLs of image files, and evaluates the intensity of relation between images and query keywords. It fetches highly evaluated images from WWW, and transfers them to an IAP in the same slave PC.

An IAP receives images from ICPs, and extracts image features from the images. Every time it receives a new image in group A, it computes distance between the new one and ones received before. In addition, for computing distances between images in different slave PCs, the system chooses a particular slave PC all of whose ICPs have been done before any other slave PC, and makes this PC receive all the images features from other slave PCs. This slave PC is called the main slave PC. The IAP in the main slave PC computes all distances between images which have been gathered by different slaves. After all ICPs terminate, the IAP sends image features and computed distance to the master PC. No images themselves are sent to the master PC so as to reduce the data volume to be sent.

5 Experimental Results

We have implemented the system on a Linux-based PC cluster, which consists of one master PC and eight slave PCs. Their CPUs are Intel Celeron 400Mhz, 450Mhz or 500Mhz, and their memory size is 256MB.

Table 1. Experimental results.

query keywords	num. of URLs	images in group A		images in group B		total (A+B)	
		collected	**selected**	collected	**selected**	collected	**selected**
lion	1363 (10)	72 (84)	**62 (93,95)**	216 (26)	**66 (42,49)**	288 (41,86)	**128 (67,73)**
apple	1418 (24)	97 (86)	**76 (95,87)**	237 (50)	**99 (72,60)**	334 (61,58)	**175 (82,71)**
baby	1746 (39)	85 (48)	**73 (53,95)**	528 (74)	**272 (83,58)**	613 (70,64)	**345 (77,62)**
desk	1280 (37)	76 (90)	**72 (92,97)**	212 (50)	**84 (71,56)**	288 (61,37)	**156 (81,72)**
keyboard	1521 (18)	39 (95)	**38 (95,97)**	167 (60)	**58 (73,43)**	206 (66,49)	**96 (82,57)**
tiger	1871 (6)	57 (71)	**51 (75,95)**	178 (33)	**71 (42,50)**	235 (42,96)	**122 (56,69)**
Nomo†	951 (5)	38 (95)	**34 (97,92)**	28 (25)	**14 (36,72)**	66 (65,100)	**48 (79,88)**
Mt.Fuji	3165 (28)	541 (71)	**317 (91,75)**	837 (42)	**158 (66,30)**	1378 (53,84)	**475 (82,53)**

†. name of a major league baseball player.

5.1 Evaluation of Gathered Images

We show experimental results for eight keywords in Table 1, which describes the number of image URLs extracted from all HTML documents, the number of images collected from WWW, and the number of selected images. Numerical values in () represent the precision and the recall of the image URLs, the collected or selected images.

In the collection part, we used five major Japanese search engines, Goo, Infoseek Japan, Lycos Japan, Ocn Navi, and Excite Japan to obtain URLs related to the keywords, and merged the search results of five engines by omitting duplications. For each keyword, we obtained about 2000 URLs of HTML documents in about ten seconds. We fetched and analyzed HTML documents, and we obtained several hundreds of images from WWW. Fetched images were divided into two groups, A and B, by analyzing HTML documents as shown in Table 1.

In the selection part, we selected images from group A by the image-feature-based clustering and removing small clusters which have fewer images than 5 percent of the number of images collected in group A, and selected images from group B by CBIR. We judged selected images either as OK or NG by the subjective evaluation. OK means that the image exactly corresponds to the keywords, and NG means that it does not. In Table 1 we describe the precision, which is defined to be $N_{OK}/(N_{OK} + N_{NG})$, and the recall, which is defined to be $N_{OK_{sel}}/N_{OK_{col}}$, where N_{OK}, N_{NG}, $N_{OK_{sel}}$, and $N_{OK_{col}}$ are the number of OK images, the number of NG images, the number of OK images in selected images, and the number of OK images in collected images, respectively. The recall only for the collected images is defined to be $N_{OK_{col}}/N_{OK_{URL}}$, where $N_{OK_{URL}}$ is the number of OK images in image URLs extracted from all HTML documents, and it is relatively high for most of the keywords. For the five keywords in the table, both the precision and the recall of images selected from group A are over 87%. This shows that most of high-scored images at the keyword-based evaluation are correct. The precision of images selected from group B is between 36% and 83%. It is superior to the precision of images collected as group B in all experiments.

As the final output of each experiment, we obtained output images the number of which was about half of the number of collected images, and the precision

is improved much compared to the precision of collected images, which are collected by only evaluation of the keywords. Especially, for targets whose color is essential to discriminate their images, for example, "apple" and "lion", we obtained better improvement. Both the precision and the recall of most of output images are about 70%, which implies that our method is effective for image-gathering from WWW.

Since Mt.Fuji is the most popular mountain in Japan, there are many images of Mt.Fuji in Japanese web sites. There are relatively fewer images of "Nomo" than images related to other keywords, since "Nomo" is a person's name. However, because most of "Nomo" images are fetched from sports news web sites and appropriate keywords are always attached to their ALT tags, their recall becomes very high.

5.2 Comparison of Execution Time

In Table 2, we compare the execution time in terms of the execution type. It shows the execution time in case of the sequential execution of the collection part and the analysis part, and as well as in case of the concurrent execution of them with zero slave PCs and six slave PCs. Note that zero slave PCs means that one PC plays both roles of a master PC and a slave PC at the same time. In this experiment, we used "lion" as a query keyword and carried out image-gathering using 1145 URLs fetched from search engines.

In the parallel execution with zero slave PCs, the speed-up is 1.11 times compared to the sequential execution with zero slave PCs. The more the number of slave PCs increases, the greater speed-up is obtained. In the experiment with six slave PCs, the minimum execution time is 65 seconds, and the speed-up is 3.63 times compared to the sequential execution time with zero slave PCs. This shows that the parallel implementation on a PC cluster is effective.

Table 2. Comparison of the processing times in the sequential execution and in the concurrent execution.

the number of slaves	execution form	execution time (seconds) (collection/analysis)	speed-up
0	sequential	236(192/44)	—
0	concurrent	213	1.11
6	sequential	107(63/44)	2.21
6	concurrent	65	3.63

Figure 4 shows the execution time, where the number of slave PCs varies from one to eight. As the number of slave PCs increases, the execution time has become shorter.

In order to evaluate the variation of the execution time according to the number of slave PCs, we define time overhead TO [9] that represents an overhead of the execution time in the experiment compared to the ideal execution time T_{ideal} by n slave PCs. It is defined as follows:

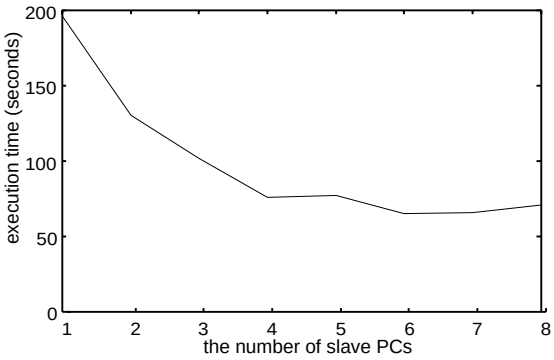

Fig. 4. Execution time according to the number of slave PCs.

$$TO = \frac{T_n}{T_{ideal}} - 1 \tag{1}$$

where T_n is the execution time by n slave PCs. If all PCs have the same CPU speed, T_{ideal} is T_1/n. Since the CPU speed of PCs used in this experiments are different, we define T_{ideal} as the harmonic mean of the execution time T_1^k on the k-th PC as follows:

$$T_{ideal} = \frac{1}{\frac{1}{T_1^1} + \frac{1}{T_1^2} + \cdots + \frac{1}{T_1^n}} \tag{2}$$

Figure 5 shows the time overhead TO, where the number of slave PCs varies from one to eight. TO is increasing nearly in proportion to the number of PCs.

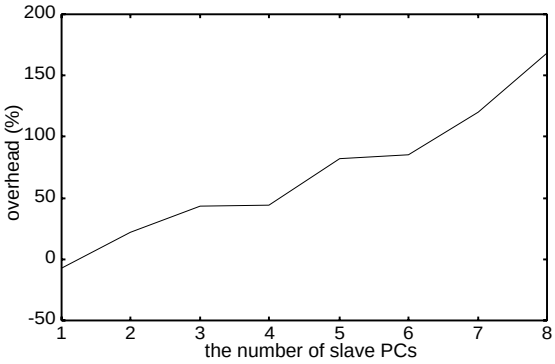

Fig. 5. Time overhead.

Many ICPs tend to be idle during the processing of the collection part, since

the number of URLs assigned for each image collection process(ICP) decreases as the number of PCs increases.

In this experiment, the fastest execution was achieved by six slave PCs. Even if we add more slave PCs, practically no more speed-up will be achieved, and the time overhead will further increase. This is due to the increase of idle processes, the increase of communication, and the limitation of bandwidth of communication lines to the Internet.

6 Conclusions

In this paper, we described design, implementation, and experiments of a fast automatic image-gathering system from WWW. We have achieved the high precision and recall that are about 70% without any knowledge about target images by means of both the keyword-based selection and the image-feature-based selection. The only input we have to supply to the system is a list of query keywords. Furthermore, we have achieved fast image-gathering from WWW by implementing the system as a parallel system on a PC cluster.

In the current implementation, we use only a color histogram as an image feature for image-selecting. For future work, we plan to exploit textures and edges as image features and integrate word histograms of HTML documents with image features. Some parameters for thresholds used in the system are given by hand at present, we plan to decide them by learning.

Acknowledgments

A part of this work was supported by a grant from the Okawa Foundation for Information and Telecommunications.

References

1. V.N. Gudivada and V.V. Raghavan, "Content-based image retrieval-systems," *IEEE Computer*, vol. 28, no. 9, pp. 18–22, 1995.
2. A. D. Bimbo, *Visual Information Retrieval*, Morgan Kaufmann, 1999.
3. C. Framkel, M.J. Swain, and V. Athitsos, "Webseer: An image search engine for the world wide web," Tech. Rep. TR-96-14, University of Chicago, 1996.
4. J. Smith and S.F. Chang, "Visually searching the web for content," *IEEE Multimedia*, vol. 4, no. 3, pp. 12–20, 1997.
5. S. Sclaroff, M. LaCascia, S. Sethi, and L. Taycher, "Unifying textual and visual cues for content-based image retrieval on the world wide web," *Computer Vision and Image Understanding*, vol. 75, no. 1/2, pp. 86–98, 1999.
6. M.J. Swain and D.H. Ballard, "Color indexing," *International Journal of Computer Vision*, vol. 7, no. 1, pp. 11–32, November 1991.
7. U. Gargi and R. Kasturi, "An evaluation of color histogram based methods in video indexing," in *International Workshop on Image Databases and Multimedia Search*, 1996, pp. 75–82.
8. J. Hafner, H.S. Sawhney, W. Equitz, M. Flickner, and W. Niblack, "Efficient color histogram indexing for quadratic form distance functions," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 17, no. 7, pp. 729–736, 1995.
9. T. A. Marsland and F. Popowich, "Parallel game-tree search," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 7, no. 4, pp. 442–452, 1985.