Proc. VIIth Digital Image Computing: Techniques and Applications, Sun C., Talbot H., Ourselin S. and Adriaansen T. (Eds.), 10-12 Dec. 2003, Sydney

Heuristics for Image Retrieval Using Spatial Configurations*

W. F. Smyth[1,3], C. P. Lam[2], Xin Chen[4], and Valerie Maxville[2]

[1] Algorithms Research Group, Department of Computing & Software
McMaster University, Hamilton, ON L8S 4K1, Canada
smyth@mcmaster.ca
www.cas.mcmaster.ca/cas/research/groups.shtml

[2] School of Computer and Information Science, Edith Cowan University
2 Bradford Street, Mount Lawley WA 6050, Australia
c.lam@ecu.edu.au

[3] School of Computing, Curtin University, GPO Box U1987
Perth WA 6845, Australia

[4] IBM Toronto Laboratory, 8200 Warden Avenue, Markham, ON L6G 1C7, Canada
axchen@ca.ibm.com

Abstract. This paper describes a strategy for content-based image retrieval via spatial configuration of objects within an image. The proposed strategy locates "pattern" images within "text" images, where the objects in each image are specified only in terms of their pairwise spatial relationships. Thus both pattern and text are represented by square matrices whose elements are drawn from an "alphabet" of spatial relationships. A match is found when the pattern image is identified as a "substructure" of a given text image. Experimental results relating to retrieval of images with low image content are shown. The approach can be used in situations where methods involving features such as colours and texture are inappropriate. We believe that such algorithms would find application in topological/GIS applications where the aerial/satellite images are very similar in terms of low-level image content.

Keywords: Multimedia, Imaging Applications, Content-Based Image Retrieval

1 Introduction

Large collections of digital images are being created in all sectors of life as digital information can be distributed easily and cheaply. Increasingly, it is vital to be able to retrieve digital images from collections as many more application areas

* The work of the first author was supported in part by grants from the Natural Sciences & Engineering Research Council of Canada.

such as the miltary, geographical imformation systems, entertainment, education and biomedicine are storing information digitally. The traditional approach to searching these collections involved either browsing or keyword indexing. Both of these two methods are labour-intensive, especially for a large collection.

Three factors characterise most current approaches to Content-Based Image Retrieval (CBIR) in terms of the content of images: low-level image features, level of abstraction and domain dependence. Most of the associated techniques employ image-processing techniques to extract semantic features within the images. Image features used for CBIR fall into two categories: low-level features and high-level features which are often calculated using the low-level features. Low-level features that have been commonly used include color, texture, edges/local orientation, wavelet transform [12] and high-level features may involve curvature, shape [7], spatial locations of objects or regions [6] as well as spatial relationships between objects/regions [10]. Some of the high-level features are also domain-specific: the representation is designed for a specific domain and often requires human intervention. Examples of the use of domain-specific high-level features include faces as in Photobook [8], trademarks [4] and attributed relational graphs [3]. The limitations associated with these approaches are due to the lack of reliable feature-extraction techniques as well as of an appropriate level of abstraction to represent the semantic content of the images (i.e., which kind of features to extract and the integration of extracted features to address the user's queries). The issue here relates to use of low-level visual features extracted from images for representing high-level semantic concepts used by humans.

Two recent surveys on CBIR systems demonstrated that most existing systems predominantly use colour, texture and shape for representing image content and for indexing and retrieval of images. From a total of 48 CBIR systems surveyed, only about eight support queries based on the spatial configurations of objects/regions in the images (i.e., find images with objects in the spatial configuration indicated in an example image). In addition, existing work involving spatial relationships usually incorporates the identified pairwise spatial relations between two objects into the indexing structure that is subsequently used for retrieval. Petrakis [9] uses the properties of each object as part of the label of a corresponding node in a graph and the pairwise relations become part of the label of an edge in the graph joining two nodes. Thus each image turns into a graph whose number of nodes is equal to the number of identified objects in the image. Lee *et al* [5] computes a signature for each image in the image database. The technique requires objects in the image to be identified and then coded with an identifier. The identifiers associated with the objects and the pairwise spatial relations between the identified objects are used in computing a signature for the image which will be used subsequently to aid the retrieval of possible matches.

Although many of the systems and techniques discussed above are effective in various application domains, they are not very effective when it comes to retrieval of images with low-level image content such as those in topological image databases (e.g. black and white aerial and satellite images). The most distinguishing characteristics associated with these images are the spatial con-

figuration of the objects within the images and the shapes of these objects. Some existing work on these kinds of images has involved retrieval via shape [1].

This paper describes an approach to image retrieval that is based solely on the spatial configuration of the objects or regions in the image. The algorithms retrieve all those "text" images (matches from the image database) that contain configurations identical to the configuration that occurs in the "pattern" image (query). Unlike the approach of Lee *et al* [5], the proposed method treats all objects as equivalent and so does not depend on any of their other features; thus the nature of the object does not need to be specified by human intervention. In avoiding the need for precise feature identification of objects, the approach is less dependent on robust segmentation techniques (as it is known that segmentation and identification of objects in images are difficult). The exact size of the bounding boxes is not important in the proposed approach — the key is the relative positions of these objects. Thus even if a bounding box has not enclosed an object totally, the approach will still work. Further, the method described here can be used with *any* system that allows the spatial relationship between any two objects to be specified.

2 Overview of Heuristic Algorithms

In this section we provide an overview of the heuristics employed in the algorithms that recognize 2D spatial patterns in a 2D "text". Due to space limitations we omit detailed explanations; these can however be found in a more complete write-up in Chen's thesis [2].

2.1 Matrix Representation of Spatial Relationships

Basic to our approach is the use of some system for representing the spatial relationship between pairs of objects in the plane. There are three such systems in common use [2] that we call ***Type 0***, ***Type 1***, ***Type 2*** in increasing order of precision and sophistication.

There are only five Type 0 spatial relationships between objects: "disjoint" (d), "partial overlap" (p), "contain" (c), its converse "belong" (b), and "edge-aligned" (e). Thus, for example, the image shown in Figure 1 gives rise to the Type 0 matrix shown in Figure 2(a): objects A–D are represented by rows/columns 0–3 in the matrix T_0. $T_0[0,1] = T_0[1,0] = p$ indicates partial overlap between objects A and B; $T_0[2,3] = T_0[3,2] = d$ tells us that C and D are disjoint.

As described in Chen's thesis [2], this very simple system is made more precise by using the 16 **Type 1** spatial relations that are derived from the points of the compass (N,S,E,W); still more precise by using the 169 (or 13^2) **Type 2** spatial relations that arise from considering the 13 possible relations between two objects in each of two dimensions. The matrices derived from the sample image using Types 1 and 2 are shown in Figures 2(b) and 2(c), respectively. The main point is that, whichever system is employed, the spatial relationships as expressed in the matrices can be represented in terms of an integer ***alphabet*** for each type

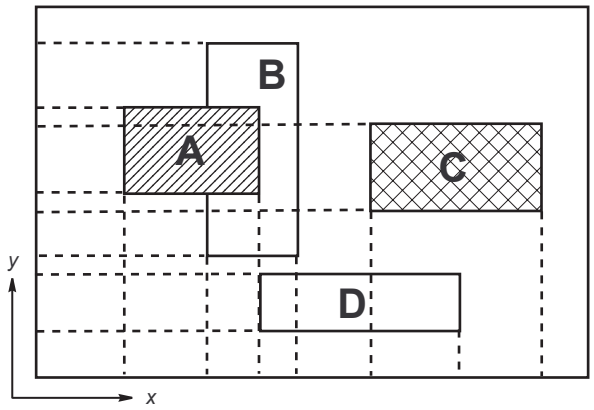

Fig. 1. A sample image.

	0	1	2	3
0	0	*p*	*d*	*d*
1	*p*	0	*d*	*d*
2	*d*	*d*	0	*d*
3	*d*	*d*	*d*	0

(a) T_0

	0	1	2	3
0	0	5	7	7
1	12	0	8	7
2	10	9	0	11
3	10	10	6	0

(b) T_1

	0	1	2	3
0	0	35	11	26
1	135	0	5	39
2	159	165	0	143
3	144	131	27	0

(c) T_2

Fig. 2. Using a 2D matrix to represent an image.

of matching. In the case of this paper, the integer alphabet for each type of matching is: 0..5 for Type 0, 0..16 for Type 1, 0..169 for Type 2.

In the language of ***string processing*** [11], our problem is to locate a certain ***pattern*** — that is, a collection of m objects arranged in the plane — as a configuration within a given ***text*** — another collection of $n \geq m$ objects in the plane. The technique we have described above allows us to formulate this problem in terms of pattern matrices $P = P[0..m-1, 0..m-1]$ and text matrices $T = T[0..n-1, 0..n-1]$. Here ***pattern*** refers to arrangement of objects in the query image and ***text*** refers to arrangement of objects in the images in the image database.

2.2 Patterns as Substructures of the Text

The problem we need to solve is illustrated in Figure 3: we need to find occurrences of P in T. But this is not a simple matter of finding P as a submatrix of

T: in fact P occurs three times in T: as $T[0..2, 0..2]$, as T with rows/columns 2 and 4 removed, and as $T[2..4, 2..4]$. Observe that in the latter case we need to be able to recognize that interchanging rows and columns in $T[2..4, 2..4]$ yields P.

P

0	0	1	2
1	3	0	4
2	5	1	0

T

0	0	1	2	2	1
1	3	0	4	4	5
2	5	1	0	3	4
3	5	1	1	0	2
4	2	3	1	5	0

Fig. 3. Example of a pattern matrix and a text matrix.

We express this requirement formally as follows:

Definition. Given a matrix $T[0..n{-}1, 0..n{-}1]$ and a sequence $I = \{i_1, i_2, ..., i_m\}$ of $m \leq n$ increasing nonnegative integers, $i_m \leq n-1$, the intersections of the m rows and m columns of T specified by I form a new matrix $P'[0..m{-}1, 0..m{-}1]$ called a ***substructure*** of T. A substructure P' of T is a ***match*** for P if there exists a permutation function $\Pi : 0..m{-}1 \rightarrow 0..m{-}1$ such that

$$P'\big[\Pi[i], \Pi[j]\big] = P[i, j]$$

for every $i, j \in 0..m{-}1$.

Finding all substructures in T that match P is a generalization of the subgraph isomorphism problem, known to be NP-complete: subgraph isomorphism corresponds to the use of an alphabet $\{0, 1\}$ for the elements of the matrices P and T. In fact, our heuristics are fast in practice precisely because we use larger alphabets; in particular, they are faster for matrices constructed using the Type 2 alphabet than for those on the Type 0 alphabet.

2.3 Algorithmic Stages

We have designed four algorithms to compute substructures, each consisting of three main stages, briefly described below. Details of these four algorithms are given in [2].

Stage 1: The Preprocessing Stage
Here we compute ***row signatures*** for both P and T, determining for each row the number of occurrences of each letter in the alphabet. If for some letter the number of occurrences in row j of P exceeds those in row i of T, we know that rows i and j are incompatible. Thus a comparison of the row signatures for P and T enables us to form a list for each $j \in 0..m{-}1$ of rows of T that could

possibly correspond to j in some match of P. The time requirement for this stage is $O\big(\max\{n, \sigma m\}n\big\}$, where σ is the alphabet size.

Stage 2: The Filtering Stage
Here we use two main strategies to eliminate, as quickly as possible, sequences of rows in T that could not possibly match P: a tree pruning strategy and a strategy that involves computation of column signatures in addition to row signatures. The time requirement for this stage is exponential in n in the worst case, but in fact is very fast in practice.

Stage 3: The Checking Stage
In this stage the sequences of rows in T that could possibly match P are checked, again using two variant strategies: one that simply does a direct check, another that precedes direct checking with another filtering mechanism, this time based on the fact that the sequence of columns must match the sequence of rows. Direct checking requires $O(m^2)$ time for each of the at most n rows checked, but again executes very quickly in practice.

The steps of one of the four algorithms are given below:

Algorithm 1
- Stage I:
  - Compute row signatures for the pattern and the text.
  - Compare two row signatures to get the lists corresponding to every row of the pattern.
- Stage II: Select feasible row sequences.
- Stage III: Direct Checking.

3 Experimental Results

Experiments for retrieval involved a set of 57 black and white images of different house plans. Things of interest on each image (house plan) were marked with bounding boxes (e.g. Figure 4) which can be seen as thick lines.

Figure 5 shows the pattern image that defines a spatial configuration of rooms and an example of one of the two retrieved image based on Type 2 matching is shown in figure 6. It can be seen that the house plan in figure 6 is the same as that shown in Figure 4 where four rooms have been marked: Bedroom 2, 3, 4 and Master. The match as shown in figure 6 includes only Bedrooms 3, 4 and Master. There is no match for the configuration of bedroom 2, 3 and 4 because as shown in figure 5, there is a part of the box at the bottom of this figure that has an overlapped section with the box on the top right hand corner of the figure.

Figures 7 and 8 shows the pattern and one of the retrieved images using Type 2 matching in another query where the intent is to find house plans where the bathroom has a specific configuration as indicated in the pattern. Here both bathrooms within the retrieved image (adjacent to Bedrooms Master and 3) were marked. Only the configuration for the bathroom of the Master bedroom was returned as a match.

Fig. 4. Example image where objects of interest are marked via bounding boxes.

4 Conclusion

In this paper we have described strategies for locating "pattern" images within "text" images, where the objects in each image are specified only in terms of their pairwise spatial relationships. Thus both pattern and text are represented by square matrices whose elements are drawn from an "alphabet" of spatial relationships. A match is found when the pattern image is identified as a "substructure" of a given text image. For an alphabet of size 2, this pattern-matching is equivalent to the NP-hard subgraph isomorphism problem, but we develop filtering strategies that in practice, for larger alphabets, allow us to process realistic text images containing 50 or so objects in milliseconds.

We have shown retrieval results relating to images with low image content where methods involving features such as colours and texture will not be appropriate. We believe that such algorithms used in conjunction with shape recognition algorithms would find application in topological applications where aerial/satellite images are very similar in terms of low-level image content.

References

1. P. Agouris, J. Carswell, and A. Stefanidis. An environment for content-based image retrieval from large spatial databases. *ISPRS Journal of Photogrammetry*

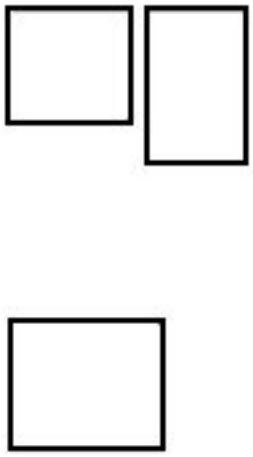

Fig. 5. Pattern image defining a specific spatial configuration of the bedrooms within a house.

and Remote Sensing, 54(4):263–272, 1999.

2. X. Chen. *Retrieval by Spatial Configuration.* M.Sc. thesis, McMaster University, 2001.

   `http://www.scis.ecu.edu.au/research/se/docs/TRSpatial-CDIR.pdf`

3. B. Huet, A. D.J. Cross, and E. R. Hancock. Shape retrieval by inexact graph matching. *Proc. IEEE International Conference On Multimedia Computing And Systems (ICMCS99)*, 772–776, 1999.
4. A. Jain and A. Vailaya. Shape-based retrieval: A case study with trademark image databases. *Pattern Recognition*, 31(9):1369–1390, 1998.
5. S. Y. Lee, M. C. Yang, and J.W. Chen. Signature file as spatial filter for iconic image database. *Journal of Visual Languages and Computing*, 3(4):373–397, 1992.
6. W. Y. Ma and B. S. Manjunath. Netra: A toolbox for navigating large image databases. *Multimedia Systems*, 7(3):184–198, 1999.
7. F. Mokhtarian, S. Abbasi, and J. Kittler. Efficient and robust retrieval by shape content through curvature scale space. *Proc. International Workshop on Image Databases and MultiMedia Search*, 35–42, 1996.
8. A. Pentland, R. Picard, and S. Sclaroff. Photobook: Content-based manipulation of image databases. *Proc. SPIE — Storage and Retrieval for Image and Video Databases II*, 34–47, 1994.
9. E. Petrakis, C. Faloutsos, and K.-I. Lin. Imagemap: An image indexing method based on spatial similarity. *IEEE Trans. on Knowledge and Data Engineering*, 14(5):979–987, 2002.
10. J. R. Smith and S. F. Chang. Visualseek: A fully automated content-based image query system. *ACM Multimedia*, 87–98, 1996.
11. W. F. Smyth. *Computing Patterns in Strings.* Pearson Addison-Wesley, 2003.
12. R. Veltkamp and M. Tanase. Content-based image retrieval systems: A survey. *Technical Report UU-CS-2000-34*, 2002.

Fig. 6. One of the retrieved image for the spatial configuration defined in Figure 5.

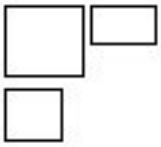

Fig. 7. Pattern image defining a specific spatial configuration of the bathroom.

Fig. 8. One of the retrieved image for the spatial configuration defined in Figure 7.