

Contour-based image retrieval in a graphic database

G. Iannizzotto, L. Vita
Facolta' di Ingegneria- Universita' di Messina
Contrada Papardo, MESSINA - ITALY -
E-mail: vita@mat520.unime.it

Abstract: The paper proposes a fast, efficient method for access to databases of images which are indexed on the basis of graphic information alone and are automatically extracted in the form of geometrical primitives. A search can be started up on even a summary graphic description of the scene, which the user provides through a simple interface similar to those used in computer graphics programs. The results obtained are satisfactory, even in the presence of images with poor contrast or of bad quality in general.

Introduction.

The growing interest in multimedia applications which has become evident over the last few years in all sectors of information science has revealed a need for new techniques for the storing and retrieval of information which, in such applications, comes in various forms - images, text, sound and speech. Although the problem of storing heterogenous information is not new, the need for the user to be able to access it with the speed typical of multimedia applications certainly is; this is the scope of the present paper, which proposes an efficient technique for the storing and retrieval of images.

An image database is substantially different from traditional alphanumeric archives, not only on account of the form of the data and the way in which it is stored, but also - and above all - because the search key (whether it be simple or multiple) is generally not homogeneous with the information stored. In other terms, whereas in current databases containing alphanumeric data the single records are sought by (simple or multiple) keys which are also alphanumeric, image databases can accept both alphanumeric codes and simple descriptions, in the form of key words, of the contents of the image being sought; these key words are not, of course, homogeneous with the data stored.

However, although in many cases a description in terms of key words is acceptable as a search key for an image, it often involves problems. An example is the fact that, as key words are linked to recognition of the main objects in the scene and, at times, even to interpretation of the scene itself, they have to be inserted manually by a human operator after careful analysis of the image. This analysis cannot overlook anything, because an important detail which the operator may not have taken into due consideration may lead to incorrect search results. Likewise, if the operator wrongly considers a detail of the scene to be important, he will insert it as a reference in the search key and thus cause the image to appear in all the searches involving that detail, even though this may not be desirable in the case at hand.

Other problems are due to the fact that a salient detail of the image can be verbally represented in several ways (i.e. with several synonymous or similar words) and this leads to a further effort to standardize the vocabulary of key words and the syntax with which they have to be composed to describe the scene in the search key.

No less important is the fact that manual insertion of search keys, especially in view of the extreme care that has to be taken in loading the database, obviously requires a very long time, often far outlasting the user's requirements.

In the light of these and other problems, it seems clear that there is a need to develop a method for describing images in such a way that it is possible to compose search keys which are compact, as homogeneous as possible with the graphic origin of the information to be stored, and can be inserted both automatically, when the database is loaded, and manually, through a suitable user interface, in the search phase.

For medium-size databases it is possible to keep an index of the images stored, containing all the keys and structured in such a way as to allow for greatly reduced search times, with all the obvious advantages this provides.

Geometrical description.

A user who has to describe an image to an interlocutor can generally use one of two methods. The first one is exclusively verbal and consists of giving a list of objects and characteristics belonging to the scene to be described; it often provides a partial interpretation of the scene and necessarily presupposes that the describer and the interlocutor are capable of recognising all the objects mentioned. This method entails almost unsurmountable difficulties if the two subjects do not speak the same language or have different cognitive backgrounds and so cannot understand each other as regards the name or even the nature of the objects in the scene.

The second method, which is universally recognised as the most efficient for communication between thinking human beings, whatever their cultural background, consists of a graphic description of the scene: a few signs on a piece of paper are often sufficient to describe an object or a whole scene. The superiority of this kind of expression derives from the fact that visual experience (and the consequent language, which is universal) is independent of both spoken and written language and, above all, that it is possible to give a simple description of the objects or characteristics of an image without necessarily knowing the nature of these objects or characteristics: although an inhabitant of Papua New Guinea may well not know the word we use for a coffeepot, it is not difficult to describe the shape and salient features of one to him graphically by means of a few geometrical primitives, in such a way that he would recognise it if he saw one. We cannot say the same for a verbal description, even if it were possible to find a common form of verbal language. These are the simple considerations we started from in our search for a simple, compact, fast representation of images.

Several years ago, basic research into image recognition and pattern analysis demonstrated that most information about the shape of an object is contained in the contours of the object; this may seem nothing new, as we know that when as children we learn to express ourselves graphically (with more or less satisfactory results from an aesthetic point of view) a pencil sketch of the outline is enough to make an object recognisable; if we dwell on a description of the points inside the outline - perhaps colouring it in and giving it a three-dimensional effect and thus greater realism by means of shading - this is only for aesthetic reasons. The image representation we are looking for can thus be confined to an outline as we want it to be as compact as possible, even at the expense of a partial loss of information.

Of the several techniques for 2D representation of the contours of an object, the one which is by nature the most compact is undoubtedly representation by segments or broken lines. Although it is rather approximate, this technique gives us all the information we need about the shape of the object, and at the same time occupies much less memory than other techniques; it is, in fact, sufficient to approximate the contours with a broken line and then memorize only the vertices of the segments making up the broken line. The method works very well with objects which do not have very jagged edges and has already been successfully used for the recognition of objects [1]. It is when the objects making up the scene have very jagged edges or are very small as compared with the size of the image as a whole that problems start to arise: approximation by segments is difficult and at times inaccurate. These problems are, however, not particularly relevant to our aims: with rare exceptions, in any scene there are dominant lines of a reasonable size as compared with that of the image as a whole, whether they be the outlines of objects, the background or lines of perspective. These lines will be correctly represented and will become part of the key used to search for the image, making it different from most other images, if not unique. This is quite enough, as any search made with more than one key may obviously give more than one result and here we are not looking for a single key.

The algorithm which follows is based on representation of the dominant lines of a scene by means of lines; the lines are made up of segments which approximate the trend of the dominant lines in a tradeoff between accuracy and the compactness of the resulting representation.

Contours extraction.

Most algorithms for edge detection look for points in the image at which there is a sharp variation in brightness) this search is usually made by calculating the brightness gradient point by point, using discrete operators such as the Sobel operator [2]. As the gradient is a vector, the result of the calculation is a sequence of pairs (one for each point) of the kind (G_x,G_y) or (G,Arg(G)), according to whether the vector is specified as a pair of components or as magnitude and phase. If we define two masks, one for the direction x and one for the direction y, a convolution of the points in the image is made, giving two values for each point - the components in the two directions of the brightness gradient. The masks for the Sobel gradient are:

-1	0	1
-2	0	2
-1	0	1

for x-axes

-1	-2	-1
0	0	0
1	2	1

for y-axes.

This formulation comes, through a brief series of passages, from the very definition of gradient vector:

$$G_x = \frac{\partial f}{\partial x} = [f(x+1, y-1) + 2 \cdot f(x+1, y) + f(x+1, y+1)] \\ - [f(x-1, y-1) + 2 \cdot f(x-1, y) + f(x-1, y+1)] \\ = (g + 2h + i) - (a + 2b + c)$$

and

$$G_y = \frac{\partial f}{\partial y} = [f(x-1, y+1) + 2 \cdot f(x, y+1) + f(x+1, y+1)] \\ - [f(x-1, y-1) + 2 \cdot f(x, y-1) + f(x+1, y-1)] \\ = (c + 2e + i) - (a + 2d + g)$$

in which we have used the letters from a to i to represent the points adjacent to the point (x,y). Using this simple notation, the 3x3 surroundings of (x,y) are indicated as:

a	b	c
d	(x,y)	e
g	h	i

It should be noted that the pixels closest to (x,y) are weighted with a factor of 2 in these particular digital definitions. Calculating the gradient over an area of 3x3 rather than using the method of the finite differences between the values of the adjacent pixels has the advantage of giving a more accurate average, thus making the gradient less sensitive to disturbances. The gradient can be defined in wider surroundings, but 3x3 operators are the most widely used on account of the tradeoff they provide between accuracy and computational requirements.

The Compass operator [3] was devised a few years ago to calculate a gradient in the form (G, Arg(G)), and is based on calculation, for each point, of the brightness gradient in well-defined, discrete directions. There are eight directions, i.e. those of the 8 neighbours of the points being examined. The masks used are:

1	1	1
0	0	0
-1	-1	-1

North

1	1	0
1	0	-1
0	-1	-1

North-West

1	0	-1
1	0	-1
1	0	-1

West

0	-1	-1
1	0	-1
1	1	0

South-West

-1	-1	-1
0	0	0
1	1	1

South

-1	-1	0
-1	0	1
0	1	1

South-East

-1	0	1
-1	0	1
-1	0	1

East

0	1	1
-1	0	1
-1	-1	0

North-East

Convolution is performed with these masks for each point in the image, the absolute value is obtained, and the greatest of the eight values obtained is considered as the magnitude of the gradient. The direction is given by the index of the corresponding mask.

Like all gradient operators, Compass respects the requirement that the module of the gradient, calculated at the same point but into two opposite directions, be the same. This causes problems in edge identification, an example of which we will give below.

Let us assume we are at the border between two objects and that the surfaces of the two objects are of a similar colour and reflectance; let us further assume that the two surfaces are not flat or, if flat, do not belong to the same plane, i.e. they form an angle other than π . In such a situation the human eye does not have to make an excessive effort to recognise the border, because if two points very close to the border and to each other but belonging to different surfaces are chosen, they will have a very similar gradient magnitude but a different direction.

In calculating the direction of the gradient, however, the two points may be assigned the same direction, as the two masks - one with the direction of the maximum magnitude and the other the opposite direction - will give the same result but with a different sign, and on account of the absolute value the results will be identical. The Compass operator is thus 'colour-blind' in such situations and cannot distinguish between the two points.

A solution can be found by abandoning the condition of symmetry: if a mask gives a negative value we force it to zero, thus distinguishing it from its symmetrical counterpart.

The result of this simple modification is that it is possible, with this operator, to distinguish with a good degree of accuracy the borders between adjacent surfaces even if they are very similar, and this is already a step forward.

After calculating the gradient point by point, the next step is mainly to search for a threshold to apply to its magnitude, in such a way as to discard all the points which have a low brightness gradient and thus extract all the probable surrounding points. The solution proposed here follows the same course and has proved to be satisfactory for our aims: a global threshold is calculated on the basis of the characteristics of the image and is therefore not determined a priori; the calculation algorithm analyzes the histogram of the magnitude values and discards a certain percentage of them, thus selecting only points which have a sufficiently high gradient.

Thinning.

In order to work properly, the subsequent algorithms need the contours to be reduced to lines, that is, only one pixel wide. We therefore looked for a Thinning algorithm which would give satisfactory results.

After a certain amount of research our choice fell on the work of Naccache and Shingal (1984) [2], which involves very short processing times and gives results of a good quality.

In general, a good thinning algorithm iteratively cancels the points of the edges of a region respecting certain basic constraints:

Neither the ends of segments or angles must be removed;

The connection of the region has to remain intact:

The region must not be excessively eroded, i.e. information about the morphology of the region should not be lost;

Single points must not be cancelled, as one of them might be all that remains of a region after the thinning process.

Without going into the details of the algorithm chosen, which can easily be found in literature, we point out that it is based on a series of 3x3 masks with which each point in the image is iteratively compared along with its 8 neighbours. The points which correspond to the description given by one of these masks are "useless" and can be removed. After a series of such operations, which only end when a complete cycle of comparisons has ended without any more points being eliminated, the image has been completely 'skeletonized' and can be passed on to the next stages of processing.

Hough transform[2].

We start by representing a line for a generic point $P \equiv (x,y)$ in polar coordinates, using the formula: $x \cdot \cos \vartheta + y \cdot \sin \vartheta = \rho$; this representation is very convenient because in Cartesian coordinates a vertical line has an infinite angle coefficient and a horizontal line has an infinite intercept with the x-axis.

Let us now consider the space of the parameters (ρ, θ) , which in our case is a plane: θ will vary from -90 to 90° , and ρ from $-\rho_1$ to ρ_1 , where $\rho_1 = \sqrt{2} \cdot \text{diagonal of the image}$; Let us discretize this plane,

establishing a discretization step for θ and one for ρ on the basis of our accuracy requirements: we obtain a matrix with one box for each ordered pair of parameters ρ, θ . We call these boxes accumulation cells. Each of these cells contains a counter, initially set to zero; for each point (x_i, y_i) in the image we equate the parameter θ with each of the discrete values it can assume and each time solve the equation of the line for the parameter ρ , obtaining a value which we then round off to the nearest discrete value. Each pair (ρ, θ) thus obtained identifies an accumulation cell: we then increase the corresponding counter. At the end of the cycle, which will involve all the points in the image, each accumulation cell will give the number of points belonging to the corresponding line. Our aim is to search for all the segments of the image, so at this point it will be necessary to go through all the possible lines (or perhaps only those which contain a number of points above a certain threshold) and identify the segments which belong to these lines, considering that a segment begins with a point on the line and ends when the size of the gap between two consecutive points encountered along the line is above a prefixed threshold.

To go back to the description of our work, the step after thinning is application of an algorithm of the type just described to extract all the possible segments from the image of contour points with the help of the Hough transform.

Composition of the Search Key.

The result of the operation just completed is a sequence of segments of various lengths which approximate the contours of all the objects present in the image, whether they be straight or curved. As approximation of a curve is made by means of a broken line formed by chords below arcs of the curve itself, in our case a curved edge gives rise to a sequence of small short segments. Moreover, not all the segments revealed, even if they are straight, will be significant: several may be of too small a size. As what we are looking for is a very small set of segments which are highly representative of the scene, the segments determined with the Hough transform are ordered by length and selected, only the longest being kept. The number of segments selected is n , which can be set parametrically and can be assumed to be, for example, 10.

So for each image we have extracted a significant sequence of n segments: as we shall now see, this sequence will be the database search key.

Storage and Search.

Each storage operation, and consequently each search, requires definition of an ordering relation by means of data can be ordered in the storage phase and selected in the search phase. In our case, the ordering relation is not as immediate as in the simpler case of alphanumerical keys.

A key is made up of n segments, each of which is represented by three parameters:

position of the centre;

inclination or orientation with respect to a reference axis (e.g. the horizontal one);

length.

As the user has to insert these parameters manually in the search phase, and he presumably does not know their exact value, or anyway cannot insert them graphically without a certain amount of uncertainty, we need to introduce a mechanism which will consider a confidence interval for each of them. This can be achieved quite simply (and quickly) by discretizing the intervals of the possible values for each parameter; discretization occurs at steps fixed a priori on the basis of heuristic considerations and proved to be quite efficient, giving the search algorithm a good level of selectivity despite its extreme simplicity and speed. In the future we foresee the introduction of soft computing techniques (Fuzzy logic) to manage ordering relation, and the use of neural networks for the extraction of segments.

The data structure in which the images are stored is quite simple in our implementation. Analysis and evaluation of the various possible structures presented in literature is beyond the scope of this paper, but our data structure was devised with a view to optimizing both the amount of memory occupied and access speed. The main structure is a table containing nc possible locations for the centre of the segment; the search starts from this table. Each cell in the table contains a list of possible orientations for the segment and for each orientation there is a list of possible lengths.

Of course the structure is based on dynamic memory allocation, so as to optimize occupation. In addition, whereas the search keys reside in the main memory, the images are stored in a mass memory and are only called up when the search ends and one or more of them has been identified as the object of the search. Fig. 1 illustrates the structure implemented.

For the search phase we implemented a graphic interface which allows the user to specify directly in a graphic form the key with which the search is to be made; of course partial insertion of a key is also provided for.

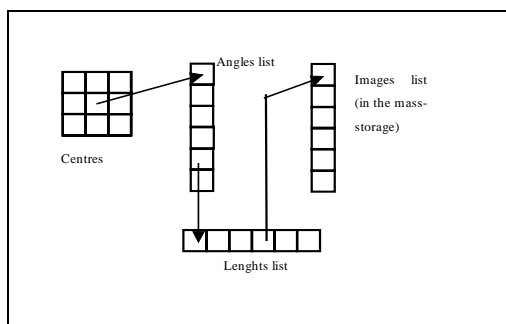


Figure 1: scheme of used data structure.

Experimental Results.

The algorithm described was written in standard ANSI C and tested by loading the file with about 100 images and carrying out a large number of searches. The results obtained were on the whole quite satisfactory, since almost every image has some typical set of straight lines (the problem is to identify this set and use it when specifying the key for the search). In general the selectivity is not such as to provide a single image as the result of a search but, within certain limits which were never exceeded throughout the tests, this is a requirement rather than a defect of the system: it is, in fact, necessary for the system to have a certain tolerance of possible (inevitable) inaccuracies in the description given by the user. Fig. 2 shows the basic steps to extract the key for an image and the graphic data introduced by the user to start the search, which gave positive results.

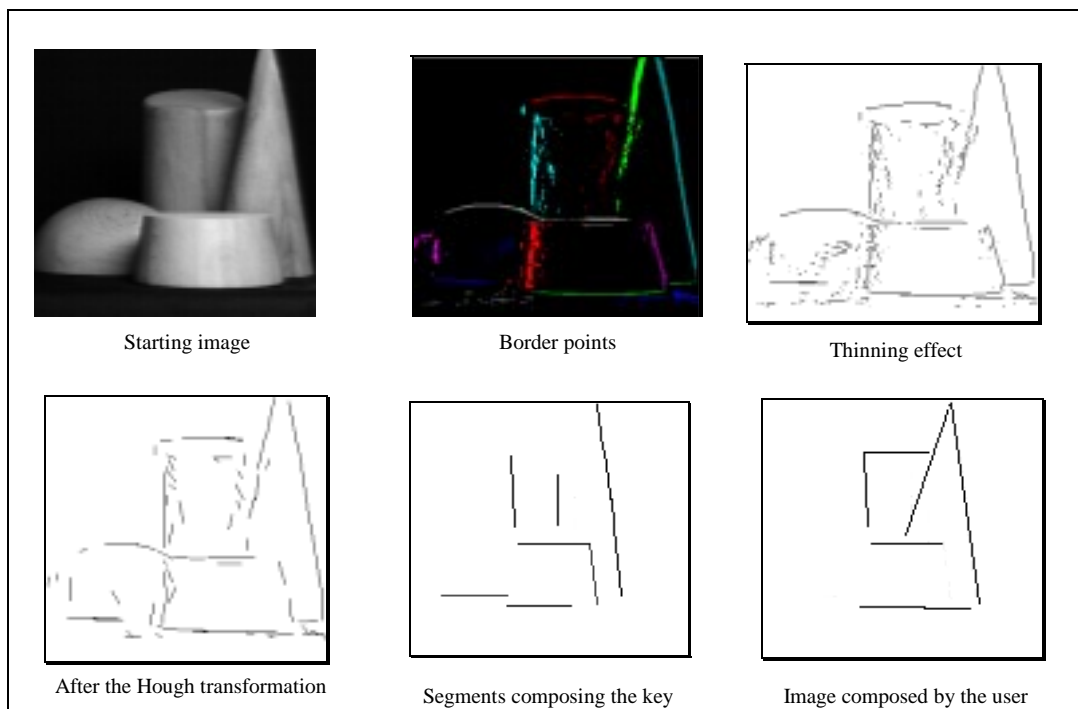


Figure 2: Experimental results

Conclusions

In this paper we have presented a complete method for image filing and search, based on automatic determination of search keys which are graphic and therefore homogeneous with the data stored.

Unlike other methods, the one presented here does not use non-graphic information (e.g. statistics on histograms of colours or levels of grey), but only geometrical primitives which describe the main contours of the objects making up the scene. The method gives a compact representation which the user can insert with great speed in the search phase. The user is not forced to choose between the various options available in terms of colour, texture, or levels of brightness, but only has to give a simple graphic description in terms of segments of the scene. Moreover, the extreme compactness of the keys obtained allows the complete index of the keys to be stored in the memory, thus speeding the search up.

The algorithms proposed to extract the contours of the objects are efficient in terms of both speed and validity of results. Although the data structures and search algorithms do not claim to be optimal, they still provide a good example of what can be obtained with this method; determination of an optimal search algorithm is beyond the scope of the paper.

Future developments will include implementation of a user-friendly user interface and porting of applications on different platforms, even of the distributed kind. Development of a fuzzy module to determine the equivalence between keys is already at an advanced stage, the aim being to enhance the selectivity of the search algorithm but at the same time preserve its current speed.

Bibliography

- [1] Sven J. Dickinson, Alex P. Pentland & A. Rosenfeld, 3-D Shape Recovery Using Distributed Aspect Matching, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol.14, No.2, Feb 1992.
- [2] King-Sun Fu, Rafael C. Gonzalez and C.S. George Lee: *Robotics: Control, Sensing, Vision and Intelligence*, Mc Graw-Hill, 1987.
- [3] William K. Pratt: *Digital Image Processing*, II Edition, J. Wiley & Sons, 1991, p. 508