Visual Database Systems, II
E. Knuth and L.M. Wegner (Editors)
Elsevier Science Publishers B.V. (North-Holland)
1992 IFIP

An Approach to Image Indexing of Documents

P.L.Stanchev, A.W.M.Smeulders, F.C.A.Groen

Department of Computer Science, University of Amsterdam
Kruislaan 403, 1098 SJ Amsterdam, The Netherlands

Abstract

The system presented in this paper supports image indexing, storage and retrieval of (paper) documents on the basis of textual or pictorial information. Examples are electronic schemas, architectural drawings and topographic maps. The system is called AMSTERDAM. We present the architecture and algorithmic background of the system, together with a software design lay-out. The main characteristics of the AMSTERDAM system are the following: (1) A semantic image representation query can be combined with a query on the basis of a pictorial specification; (2) A new methodology, based on image contents, is developed to organize and index the images; (3) New matching techniques are introduced for establishing the correspondence between the query image and the images in the database; (4) A powerful user interface, combining image processing functions together with database functions has been realized based on tabular form.

As an illustration, results of the functioning of the AMSTERDAM system in the domain of electronic schemas images are presented.

1. INTRODUCTION

While database technology has emphasized the management of textual data, in the past few years the ability to manage non-textual data has increased rapidly. This interest from the database community has manifested itself in proposals for various types of multimedia databases [1], while from the computer vision community it has manifested itself in the design of image databases [2].

We will define an image database system as a collection of images and tools for their access. Most of the existing image database systems were developed for use in specific application areas such as geographic data processing, computer-aided design, remote sensing of earth resources, medical applications and office applications. Unfortunately, most of the systems appearing under the heading of image databases are often image systems without full database functionalities or database systems not directly dealing with images. Moreover, most of the existing systems are application specific, that is, the way in which images are stored, organized and retrieved is specific for a certain application and cannot be generalized to different applications. Some valuable surveys of image databases are presented in [3-5] and some collections on papers in [2], [6].

The query of image databases is much different from the query of text databases. Various image query approaches are being described. These queries can be divided into two types of languages. The first are QBE (Query-by-Picture-Example) like languages [3], [7-8] and the second are SQL-like languages [9-10], for managing pictorial database system. An attempt to define a standard set of data manipulation operations that should be supported by generalized image database management systems is found in the high level query language PICQUERY as proposed by T. Joseph and A. Cardenas [8].

In the Amsterdam system the image indexing is defined as the search for pictorial objects in an image database on the basis of a pictorial specification. This definition goes beyond those approaches to image indexing where the pictures in the database are first reduced to a textual and numerical abstract, and the search then is restricted to searching the abstract. The pictorial component of the image indexing approach is based on hierarchically organized image detail recognizers. Fuzzy mathematical morphology Hit-or-Miss operations express the measure of correspondence of a pictorial search request with every location of the images in the database [11-12]. The fuzzy operations allow for slight discrepancies between the target image and the search request including model variation and scanner noise.

The textual component of our approach is based on a verbal description of the image content, specified whenever an image enters the database. The description is placed in a separate relational database. Textual components of the search request (currently limited to key word and numerical codes) are matched on this database using standard database search techniques. The query language in the system combines the features of QBE with SQL like languages.

In the next chapter we will present the architecture and functions of the AMSTERDAM system. After that an example for the application of the system for input, storage and retrieval of electronic schemas images will be given.

2. THE AMSTERDAM SYSTEM ARCHITECTURE AND FUNCTIONS

The AMSTERDAM system incorporates techniques borrowed from image processing, pattern recognition and databasing. The architecture of the system is shown in Fig.1.

Three levels of interaction are provided: domain definition; image entering; and image retrieval and feature extraction. The first level is the domain definition, for domain experts, who

wish to develop new application areas for the system. At the second level images may be entered into the system. The third level is image retrieval and feature extraction for the end-users who use the system only for answering specific queries.

In the system, images are indexed (i.e. obtaining the semantic image representation), before storing them into the system.

We will discuss now the process of image indexing and then the thee levels of the AMSTERDAM system.

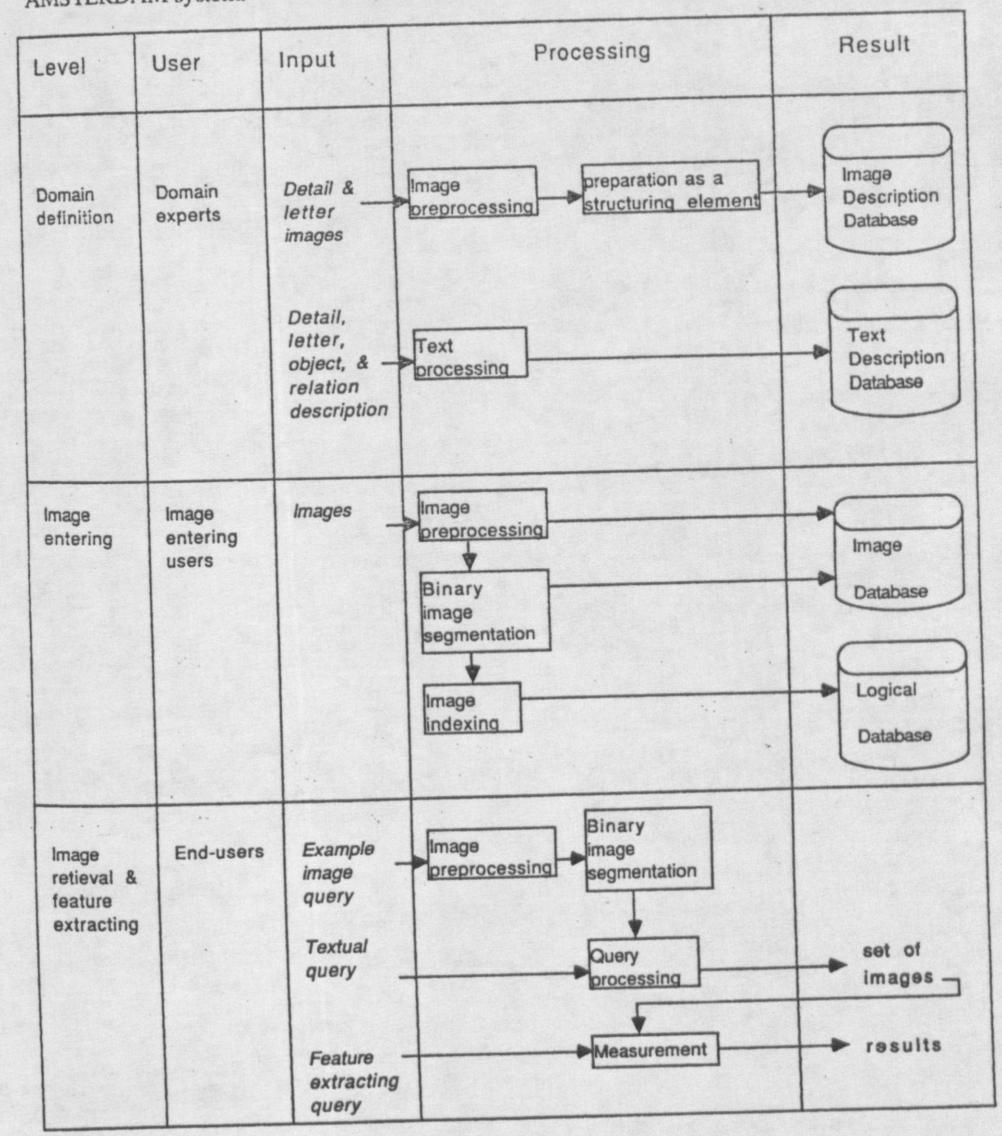

Figure 1. The architecture of the AMSTERDAM system

2.1. Image Indexing

The basis of image indexing is a semantic representation of images. The image indexing approach proposed here is based on detail, object and relation recognition.

A detail is small patch in the image with descriptive power. Details are atomic in the sense that they are the smallest image parts for which a recognition technique is applied. The best choice for an ensemble details is to form a set from which the image is composed. For geographic maps the natural set is the signs in the legend completed with the letters of the alphabet at different fonts, and standard shadings. For electronic schemas an appropriate set of details is the set of iconograms of components extended with details representing in all variations.

Details are entered by a scanner or a camera, or extracted from stored documents, and are prepared by the user with the help of some image manipulation functions. In the database, details are inserted as an image detail (which typically is anywhere from 10 x 10 to 64 x 64 picture elements of pixels) and their names. In addition, detail descriptions also include: (1) information entered from outside sources concerning the detail and (2) information extracted from the image in the topographical context of the detail. In the first case, for example, if the detail represents a legend sign for a town, the user could enter the town population. In the second case, for example, if the detail is a capacitor in an electronic schema, a number (identifying the capacitor and its value) can be found to the right of it in the image.

The semantic representation of a detail is defined as:

detail = (*detail_name*(*X*, *Y*, *att_name*, *att_value*, *add_inf*, *rec_degree*)),

where *detail_name* is the name of the detail, *X* and *Y* are the coordinates of the centre of the detail, *att_name* is the attribute name specified by the detail description, *att_value* is the information entered interactively when the detail is recognized, *add_inf* is a string of characters extracted in the topographical context of the detail specified in the detail description, and, *rec_degree* is the detail recognition degree.

An object is a semantically meaningful composition of details. Note that the difference between an object and a detail is that the latter are atomic and the former are not.

Objects are identified by their names, and by a hierarchically organized sequence of the details and the relative positions with respect to the order detail in the sequence. The object description may include: (1) information entered in interactive way, concerning the object and (2) information extracted from the image in the topographical context of the object. The semantic representation of an object is defined as:

object = (*object_name*(*X*, *Y*, *att_name*, *att_value*, *add_inf*, *rec_degree*)),

where *object_name* is the name of the object, *X* and *Y* are the coordinates of the first detail centre and the other symbols are similar to the symbols in the detail representation.

The image indexing method is based also on the relative spatial relationships elements (details or objects) that are of interest for the application. The semantic representation of relation is defined as:

relation = (*relation*(*first_element_name*, *second_element_name*)),

where *relation* takes value YES or NO, depending wherever the connection exists or not.

The total semantic representation of an image is defined as:

$d_1, d_2, ..., d_n, o_1, o_2, ..., o_m, r_1, r_2, ..., r_k$, where d_i is the detail semantic representation, o_i is the object semantic representation and r_i is the relation semantic representation.

2.2. Domain Definition

Given the semantic representation of images we now get back to the three levels of interaction as displayed in Fig.1.

The purpose of the domain definition is to describe the basic characteristics of the application domain of the images in the database. The domain definition supplies the image indexing process with initial information.

The domain description functions include: (1) detail entering; (2) letter entering; (3) object description; (4) relation description.

2.2.1. Detail entering function. The purpose of this function is to enter the details and their descriptions into the domain description. From a set example digital images details are selected one by one containing typical examples of the domain. The set of details should cover the entire pictorial variety of the domain. The image data are processed and entered in the Detail Image DataBase. The descriptions of a detail are entered in the Detail Description DataBase .

The image detail processing is done to prepare the image detail as a structuring element as known from Mathematical Morphology [11], [13] where a structuring element is a pictorial entity representing a sign used later on to search for those places in the images of the database where there is an instance of such a sign. The image is normalised before entering a detail into the database. After the normalisation positive values in the detail will indicate the image of the sign the detail expresses and the negative values correspond to the surrounding background. In addition, it is required that the sum over all elements of the normalised detail equals zero.

The detail description contains: detail name, attribute name, additional information, threshold value. The attribute name is the name of the information, which will be gathered in an interactive way from the user. The additional information describes an rectangular area near the detail in which letters will be searched. The threshold value is the sum of the positive detail image pixels.

2.2.2. Letter entering function. A special case of details are images of characters rendered in a font which is present in the domain. In the system we call them letters. We search for them in the image in an area near the elements. They are entered and processed in the same way as details are. The Letter Image DataBase includes the letter images, and their names and threshold values are stored in Letter Description DataBase.

2.2.3. Object description function. The purpose of this function is to enter the description of the objects. The description is entered in the Object Description DataBase. The description contains: object name, attribute name, additional information (similar to the information in detail descriptions). Also, all details included in the object are described by their names and relative coordinates of their centre.

2.2.4. Relation description function. The purpose of this function is to provide search capability information on the spatial relationships between elements. The description of the relation is entered in Relation Description DataBase. The expert user specifies the name of the first and second element in the relation. Then, the system will define a connection between these two elements.

2.3. Image Entering

Once the domain of the database has been defined, images can be entered into the database. These functions can be used throughout the system. The image data are stored into the Image DataBase. The images are indexed and the index descriptions are stored into the Logical DataBase. This process can be done fully automatically or interactively, if this is specified in the domain definition.

The image write function includes: (1) preprocessing, (2) search for details, (3) search for objects, (4) search for relations, (5) store the image into Image DataBase and the obtained descriptions in Logical DataBase.

2.3.1. Preprocessing. The preprocessing functions includes noise removal and it is based on [14]. Gaps in the objects in the map image are filled by closing - a dilation followed by a erosion. Then, a contrast stretching algorithm is applied to normalize the image and to correct to shading. After this preprocessing, simple thresholding is sufficient to segment the image into details, connecting lines and background.

2.3.2. Search for details. The task for the pictorial search for a specific image detail is to find instances in the image database where the same image is found as the one expressed by the detail. So, in order to find one, we need a measure which expresses the correspondence between the normalised detail image, see above, and all places in all images of the database. The theory of mathematical morphology provides a measure to express such correspondence by considering the normalised detail image as a structuring element, S, in what is called a Hit-or-Miss operation [11], [15]. The images in the database are valued +1 to indicate a black, detail pixel and -1 to indicate a white, background pixel. When, at a certain spot of an image, all positive pixels of S fit precisely in the detail, and all negative pixels fit precisely in the background, then the operation is said to yield a Hit. When either of the two fits fail, it is a Miss. It is clear that in this way instances of the detail image can be found by repeating the operation on every spot of every image in the database.

In practice, we prefer to use fuzzy mathematical morphology rather than the exact version. Fuzzy morphology permits for some inconsistency of the image with the detail or some noise in the image. At every neighbourhood of a locus (X,Y) in an image, in fuzzy mathematical morphology, the point by point weighed sum of S with the image is taken. Some pixels of S are negative and when they weight a detail value they will tend to lower the outcome. The same holds for positive values in S which hit on a background value. And these two options occur in most places where there is no correspondence between the image of the detail and the local configuration of the image of the database. But, when the local arrangement matches, then both types of pixels of S work to produce a strong positive outcome. In the ideal case, the outcome is equal to the absolute sum of all pixels in S.

Let we denote the binary version of the original image with X, then the image value is +1 for the foreground pixels (black), and -1 for the background pixels (white). A pictorial data structure, $f(x,y)$, called the structuring element contains the pictorial search specification expressed by positive and negative weights. The fuzzy Mathematical morphology Hit-or-Miss

transform can be expressed as:

$$HOM(x,y) = X \oplus_t f = \Big< \sum_{r\,s \in f} f(r,s)\; I(x-r,y-s) \Big>_t ,$$

where I is the binary image. The threshold operation on a function g, $<.>_t$, is defined by:

$$< g' >_t \quad : \quad g'(x) < t \quad \Rightarrow \quad < g' >_t = 0$$
$$g'(x) \geq t \quad \Rightarrow \quad < g' >_t = 1$$

So, the outcome of the operation will be a binary valued image with a value of +1 for those places in the image where there is a good correspondence, and a value of 0 where there is none. Consider the following example.

Let the search specification be specified by a small square box, represented as follows:

$$\text{detail} = \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix}$$

We transpose the detail in a structuring element of size 5x5 by assigning zero values to the background and assigning weights, the more to the inside the higher value they receive. (Usually details are represented by much larger neighbourhoods, but the 3x3 size suits the illustration best):

$$f(x,y) = \begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 2 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$

where an "0" indicates that the value of the corresponding pixel in the image does not care, and a higher positive value assign more weight to the corresponding pixel in the image. The corresponding threshold value for such a detail can be sensibly put at 80%, in this case the value of 8.

Let a portion of an arbitrary image in the database look like this:

.	.	4	3	3	2	2	3	.	.
.	5	3	3	2	2	3	5	4	.
.	6	2	2	2	3	3	7	2	6
5	3	3	2	3	2	3	4	3	7
3	2	2	3	1	1	3	2	4	5
2	2	3	1	1	1	3	3	2	2
3	3	2	2	1	1	2	3	2	3
2	3	2	2	1	1	3	3	2	.
.	.	2	1	1	1	1	2	.	.
.	.	.	2	2	3	2	3	3	.

Then, first it is transposed into a binary image by some segmentation operation:

.	.	-1	-1	-1	-1	-1	-1	.	.
.	-1	-1	-1	-1	-1	-1	-1	-1	.
.	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	+1	-1	-1	-1	-1
-1	-1	-1	-1	-1	+1	-1	-1	-1	-1
-1	-1	-1	-1	+1	+1	+1	-1	-1	.
.	.	-1	+1	+1	+1	+1	-1	.	.
.	.	.	-1	-1	-1	-1	-1	-1	.

Now, we compute the weighed sum for every location in the image. We show one location:

.	.	-1	-1	-1	-1	-1	-1	.	.
.	-1	-1	-1	-1	-1	-1	-1	-1	.
.	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	0.-1	0.-1	0.-1	0.-1	0.-1	-1	-1
-1	-1	-1	0.-1	1.-1	1.+1	1.-1	0.-1	-1	-1
-1	-1	-1	0.-1	1.-1	2.+1	1.-1	0.-1	-1	-1
-1	-1	-1	0.-1	1.+1	1.+1	1.+1	0.-1	-1	-1
.	.	-1	0.+1	0.+1	0.+1	0.+1	0.-1	.	.
.	.	.	-1	-1	-1	-1	-1	-1	.

which gives for that location a correspondence value of 6, under the threshold value of 8. So after the operation there is only one location in this image where the correspondence is sufficiently close. That location in the bottom of the picture is indicated in the result as a Hit.

The pixels of high correspondence in a rectangular of the same size of the detail are reduced to one hit, because the fuzziness of the operation acts as an adoptive noise filter introducing more than one hit in images of good quality.

2.3.3. Search for objects. After having found the details, the system is able to recognize objects as compositions of details. The following procedure is used: if the first detail is found the expected positions of other details are calculated using the object description. Some uncertainty in the position is allowed, by searching in a circle of radius p around the expected position. In the system the value of p is specified by the user in the stage of domain definition.

We make the assumption that noise is evenly distributed in the image. In this case the recognition degree of object can defined as:

$$\frac{\sum \text{detail recognition degree} * \text{detail area}}{\sum \text{detail area}}.$$

2.3.4. Search for relations. The relation description process includes searching for

connections between elements. In the binary image the elements found are isolated and the interconnections between them are labelled.

2.3.5. Store. Both the original image and the binary image, obtained after the preprocessing, are stored in the Image DataBase. The Logical DataBase includes: Detail, Object and Relation DataBases. Also index containing the specific features of the elements are generated [16].

2.4. Image Retrieval and Feature Extracting

The image query language in the system is a combination of retrieval by examplatory image and by textual description of the image content. When a user's request can be expressed in terms of the extracted image description, there is no need to retrieve and process the original images. If however, the textual information is not sufficient, all images should be processed at the picture level to compare them with the image example.

Since the image retrieval by content is not an exact process (there is no exact way of defining the image content), and even the user may forget to specify essential characteristics of the required search image, combining image retrieval by example with textual search is an essential moment in our system. It decreases the number of false ends.

2.4.1. Retrieval by example image. An example image is entered by scanner or camera or interactively extracted from stored images (typically the ones used by the experts to define the domain). The image query algorithm includes: (1) example image preprocessing; (2) indexing the example image (i.e. transform the visually presented query in semantic image description); (3) translation in text like query (i.e. include in the query the relation between elements); (4) binary search in index tables in Logical DataBase; (5) binary search in Logical DataBase; (6) comparison between the example image and the set of extracted images from Image DataBase.

The resemblance degree between example image and the images is defined as the recognition degree in fuzzy Hit-or-Miss transform.

2.4.2. Retrieval by image content. The user specifies an image retrieval by conditions in the form: RETRIEVE IMAGES *<image_clause>*.

The *<image_clause>* contains *<element_clause>* and *<relation_clause>*. The *<element_clause>* includes combination of *<element description>* and *<position description>*. The *<element description>* includes: element name; attribute name; conditions to the attribute values such as comparing with "=", ">=", ">", "<", "<=" ,"<>" relations for numerical value, and with *MATCH* operation for string value; and additional information, associated to the elements. The *<position description>*s are *"E"*, *"W"*, *"N"*, *"S"*, *"NW"*, *"SW"*, *"NE"*, *"SE"*, where *"E"*, *"W"*, *"N"*, *"S"*, denote the four cardinal points. The position description is interpreted from left to right not permitting brackets. The *<relation_clause>* expresses existence, nonexistence or indifference.

The image query algorithm includes binary search for images fulfilling the query in Logical DataBase and in the corresponding index tables. The resemblance degree of recognition between the query and the found images is defined as:

$$\frac{\sum \text{element degree recognition}}{\text{number of elements}}.$$

2.4.3. Feature extracting. These operations are designed to perform measurements of an element by pointing at their images.

The user points at an element of special interest to initiate a measurement. The following measurement functions are available in the system: *AREA* - the area of an element; *ANGLE* - the angle of an element; *CR* - the contour ration; *GRAVX* - the X coordinate of the moment of gravity; *GRAVY* - the Y coordinate of the moment of gravity; *HEIGHT* - the height of an element enclosing rectangle, *PER* - the perimeter of an element; *WIDTH* the width of an element enclosing rectangle; *XMAX* - the maximum value of X element coordinate; *YMAX* - the maximum of Y element coordinate, *XMIN* - the minimum value of X element coordinate; *YMIN* - the minimum of Y element coordinates.

3. AN EXAMPLE

The AMSTERDAM system manipulation capabilities are illustrated in the following example for the application of electronic schemas.

3.1. Domain Definition

3.1.1. Detail entering function. All details and the corresponding description are pointed out one by one. The details are entered with the detail write function. The entering of the capacitor image is shown in Fig. 2. The detail read function shows the contents of the Detail Description DataBase. Part of this DataBase is shown in Fig. 3. Some of the details images are given in Fig. 4., in the way they are stored in Detail Description DataBase.

3.1.2. Object description function. At this point rules for object recognition are to be defined. They are entered with the object write function. Fig. 5. shows the object read function and Fig. 6. illustrates the object write function for a part of the objects.

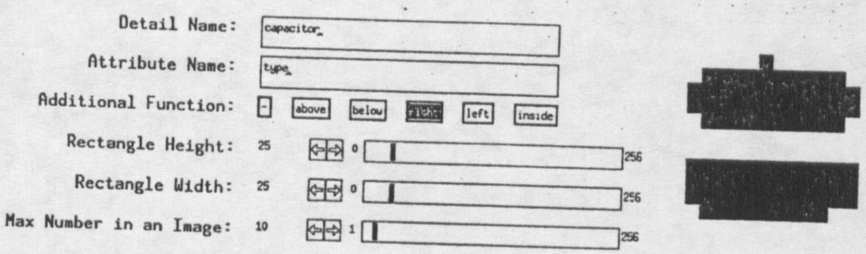

Figure 2. Detail write function.

No.	detail_name	attribute_name		add_inf		t_val	m_num
. 1	resistor_a	type	above(	35,	20)	5133	10
. 2	part_b	-	- (	0,	0)	4145	5
. 3	resistor_b	type	right(	25,	25)	6048	10
. 4	part_a	-	- (	0,	0)	6134	10
. 5	part_c	-	- (	0,	0)	6001	10
. 6	capacitor	type	right(	25,	25)	10999	10

Figure 3. Detail read function.

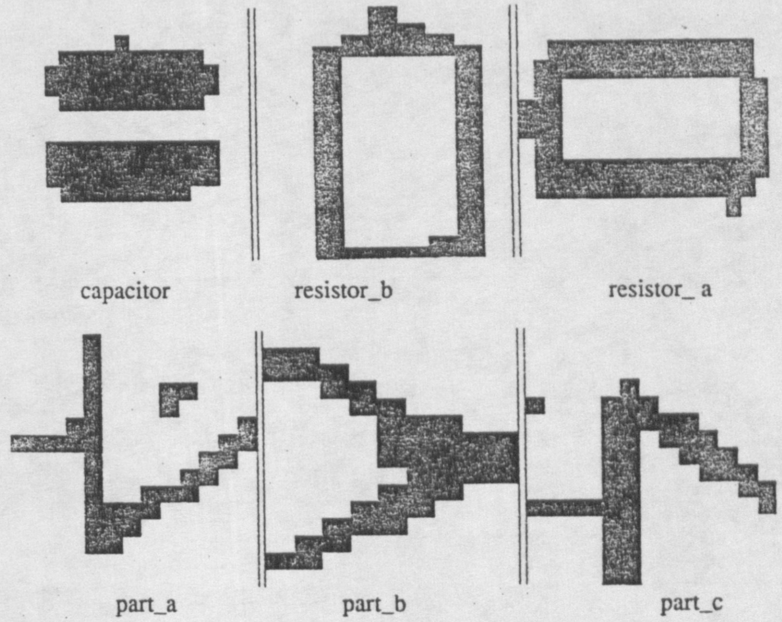

Figure 4. Part of the details into the Detail Image DataBase.

No.	obj_name	obj_name		add_inf		det_name	X	Y
. 1	opamp	type	inside(	45,	45)	part_a	0	0
. 2	-	-	- (	0,	0)	part_b	17	-12
. 3	-	-	- (	0,	0)	part_c	0	-27

Figure 5. Object read function.

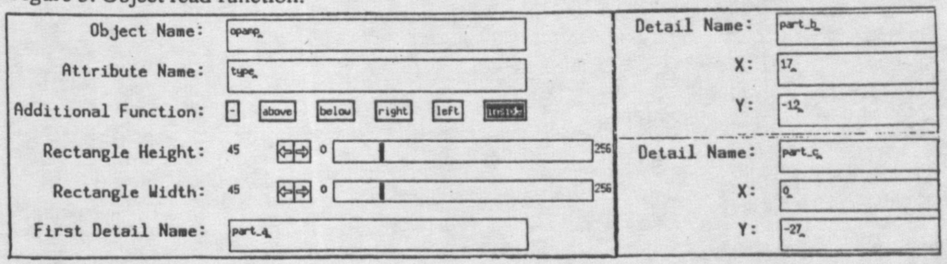

Figure 6. Object write function.

3.1.3. Relation description. In case the user wants to search for a direct connection between a resistor of type resistor_a and a capacitor, and also for a relation between two resistors of type resistor_a and _b, the corresponding relational write and read function are shown in Figs. 7. and 8.

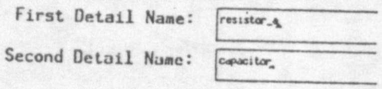

Figure 7. Relation write function

No.	1-st argument	2-nd argument
. 1	resistor_a	capacitor
. 2	resistor_b	resistor_a

Figure 8. Relation read function

3.2. Image Entering

The image write function is shown in Fig. 9.

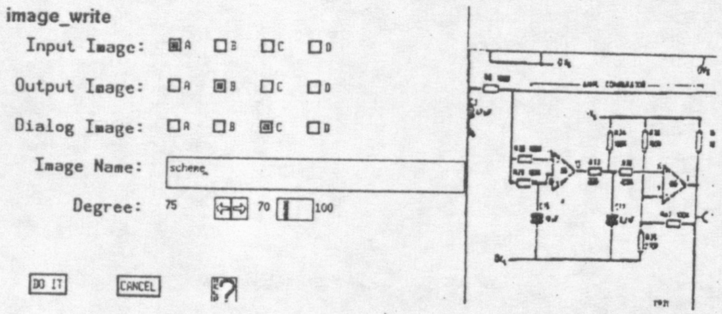

Figure 9. Image write function.

As a result of the image indexing phase, the semantic image description is obtained. Part of the information is entered in an interactive way by the user. They are stored in the Logical DataBase. Parts of this database are presented in Figs. 10, 11, and 12 for the image of Fig. 9.

No.	image_name	det_name	x	y	att_name	att_value	add_inf	degree
. 1	scheme	resistor_a	20	37	type	rr	A3	0.89
. 2	scheme	resistor_a	58	106	type	rr	A4	0.90
. 3	scheme	resistor_a	131	117	type	rr	A	0.87
. 4	scheme	resistor_a	163	117	type	rr	A	0.87
. 5	scheme	resistor_a	57	131	type	rr	A34	0.83
. 6	scheme	resistor_a	214	167	type	rr	A	0.99
. 7	scheme	resistor_b	238	82	type	rr	A	1.00
. 8	scheme	resistor_b	182	83	type	rr	A34	1.00
. 9	scheme	resistor_b	146	85	type	rr	A23	0.96
. 10	scheme	resistor_b	181	183	type	rr	234	0.95
. 11	scheme	capacitor	150	165	type	c1	4	0.89
. 12	scheme	capacitor	72	166	type	c2	-	0.86

Figure 10. A part of a detail description.

No.	image_name	obj_name	x	y	att_name	att_value	add_inf	degree
. 1	scheme	opamp	89	131	type	aa	A3	0.89
. 2	scheme	opamp	204	142	type	aa	A3	0.93

Figure 11. A part of an object description.

No.	image_name	1-st argument	2-nd argument	Relation
. 1	scheme	resistor_a	capacitor	Yes
. 2	scheme	resistor_b	resistor_a	Yes

Figure 12. A part of a relation description.

Detail and object display function for the image in Fig. 9. are shown in Figs. 13. and 14.

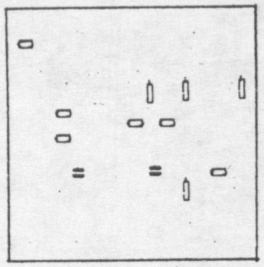

Figure 13. The detail image display function.

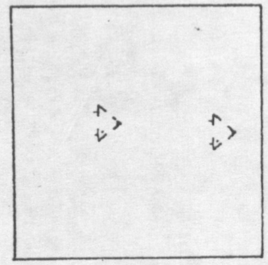

Figure 14. The object image display function.

3.3. Image Retrieval and Feature Extracting

3.3.1. Retrieval by example image. In Fig. 15. an example of the searching by example image is given.

3.3.2. Retrieval by image content. A similar retrieval but now specified as a retrieval by text, can be specified as follows:

relation(1st_element = "*resistor_b*", 2nd_element = "*capacitor*") = **INDIFFERENT**

AND

relation(1st_element = "*resistor_b*", 2nd_element = "*resistor_a*") = **INDIFFERENT**

AND

element(name = "*resistor_a*", attribute name = "*type*", operation = **MATCH**, attribute value = "*rr*", additional information = "")

NORTH

element(name = "*resistor_a*" , attribute name = "*type*", operation = **MATCH**, attribute value = "*rr*", additional information = "")

EAST

element(name = "*opamp* ", attribute name = "*type*", operation = **MATCH**, attribute value = "*aa*", additional information = "").

Figure 15. Example query for retrieval by example image.

3.3.3. Feature extracting. The way to measure the height and width of the capacitor and resistor_a is given in Fig. 16.

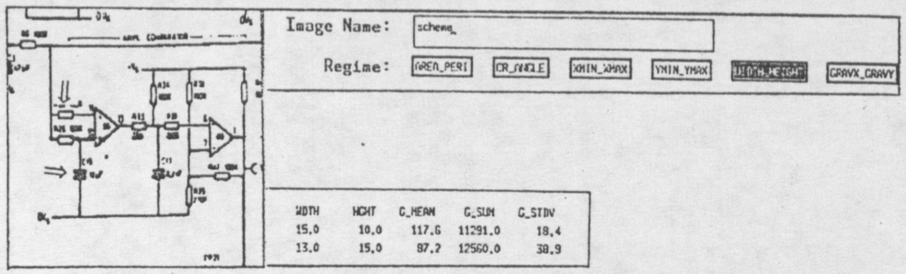

Figure 16. The measurement function.

4. CONCLUSION

The main advantages of the AMSTERDAM system for document image indexing, storage and retrieval can be summarized as follows:

- the image query can be given as an example image, as a textual query about the image content, or as a combination of both forms;
- robustness to noisy recording and inconsistencies in the indexing due to the use of fuzzy Mathematical Morphology;
- the possibility to handle different kind of images;
- fully automatically or interactive way of image indexing;
- resemblance measure calculation between the query and the retrieved image;
- simplicity of the query language;
- simplicity of the logical data representation.

Our present implementation includes all the functions described above. The system has been implemented on a SUN workstation, under UNIX and has been written in C. It uses the SCIL-IMAGE package [17], developed at the University of Amsterdam. The design of this system uses the experience of an image database systems prototype [18], working with graphical images and applied for storage and retrieval of house furnishing design drafts.

5. ACKNOWLEDGEMENT

This research was supported in part by grant: NLM - Ep (1 R01LM 05007 -01A1).

6. REFERENCES

1 E. Bertino, F. Rabitti F. and S.Gibs, "Query Processing in a Multimedia Document System", *ACM Trans. on Office Information Systems*, Vol. 6, N.1, pages 1-41, 1988.

2 W. Grosky and R. Mehrotra, Guest editors, *Computer*, Dec. 1990.

3 N. Chang and K. Fu, "Picture Query Languages for Pictorial Data-Base System", *IEEE Computer*, Vol.14, pages 23-33, Nov. 1981.

4 H. Tamura and N. Yokoya, "Image Database Systems: A Survey", *Pattern Recognition*, Vol.17, No.1, pages 29-43, 1984.

5 S. Chang, "Image Information Systems", *Proc. of the IEEE*, Vol. 73, No. 4, pages 754-764, Apr. 1985.

6 S. Iyengar and R. Kashyar, Guest editors, *IEEE Trans. on Software Engineering*, Vol. 14, No.5, May 1988.

7 N. Chang and K. Fu, "Query-by-Pictorial-Example", *IEEE Transactions on Software Engineering* , Vol. SE-6, No. 6, pages 519-524, Nov. 1980.

8 T. Joseph and F. Cardenas, "PICQUERY: A High Level Query Language for Pictorial Database Management", *IEEE Trans. on Software Engineering*, Vol. 14, No. 5, pages 630-638, May 1988.

9 N. Roussopoulos and C. Faloutsos, "An Efficient Pictorial Database System for PSQL", *IEEE Trans. on Software Engineering*, Vol. 14, No. 5, pages 639-650, May 1988.

10 P. Stanchev and F. Rabitti, "Image Database Management System. Applied Theories, Tools and Decisions", *Mathematical Research* Vol. 55, Akademie Verlag, Berlin, pages 208-214, 1989.

11 R. van den Boomgaard, "Threshold Logic and Mathematical Morphology", in *Proc. 5-th Int. Conf. on Image Analysis and Processing*, Positano, Italy, pages 111-118, 20-22 Sept. 1989.

12 P. Maragos, "Tutorial on Advances in Morphological Image Processing and Analysis", *Optical Engineering*, No. 26(7), pages 623-632, 1987.

13 A. Smeulders, "An Introduction to Image Processing and Computer Vision", *University of Amsterdam*, 1991.

14 F. Groen F and R. van Munster, "Topology based Analysis of Schematic Diagrams", *IEEE 7th Int. Conf. on Pattern Recognition*, Montreal, Canada, pages 1310-1312, 30.7 -2.8. 1984.

15 J. Serra, "Image Analysis and Mathematical Morphology", London, Academic, 1982.

16 P. Stanchev and V. Vutov, "A Model-Based Technique with a New Indexing Mechanism for Industrial Object Recognition", *IEEE/CHMT'90 IEMT Symposium*, Washington D.C., pages 56-60, 1990.

17 T. ten Kate, R. van Balen, A.W. Smeulders, F.C. Groen, G. den Boer, "SCILAIM: a Multi-level Interactive Image Processing Environment", *Pattern Recognition Letters* 11, pages 429-441, 1990.

18 F. Rabitti and P. Stanchev, "GRIM_DBMS: a GRaphical IMage DataBase Management System", in: *Visual Database Systems*, T. Kunii (Editor), Elsevier Science Publishers, pages 415-430, 1989.

(62.) Stanchev P., Smeulders A., Groen F., "An Approach to Image Indexing of Documents", "***Visual Database Systems, II***", IFIP Transaction, A-7, E.Knuth and L.Wegner (eds.), North-Holland 1992, 63-78.