Syntactical or structural pattern recognition of formal language with regular shape model : Describing the boundary of objects

Seong Dong Kim*, Ki Ho Choi** *Regular Members*

ABSTRACT

Human vision allows people to perceive and interpret the environment surrounding them. It is mostly concerned with recognition, for example, acknowledgement of other people, cars, houses or any object in general. Mechanical vision, or computer vision, is concerned with using computers to mimic the human visual system. This paper concentrates on the study of recognizing and classifying objects or patterns in digital images. In particular, it focuses on a class of pattern recognition, in which objects are recognized and classified based on the regularity of shapes. The attempt to solve the pattern of recognition in general is too far ambitious. Here, we attempt to solve a restricted family of pattern recognition problems. The aim is to develop a general approach to solving wider range problems. The main reason for not using edge detection methods is that they do not segment the objects of interest under study, objects that move.

The main objective of this paper is to investigate and develop the general approach formally from a new theory based on a unfamiliar idea, i.e, structural or syntactical pattern recognition to find formal strings for regular shape model. This approach method, which is intrinsically discrete, is as follows: line segments, horizontal line, vertical line, are used to represent and classify the regular shapes. The implementation method is used grammatical string matching for regular shapes, triangles, rectangles and rhombuses.

I. 서 론

The history of syntactic pattern recognition began in the middle 1960's. when the learning machine concept was the main issue pattern recognition. Automatic feature derivations was required to automatic the inference of a classifier from the raw data. Several statistical inference techniques were developed to overcome this problem, but soon the complexity of numerical feature derivation became evident. More complex feature representations were required to represent the components of the objects in an image, which led to introduction of symbolic features and their spatial relationship.

These concepts were introduced by R.Kirsch, R.Ledley, Naraarimhan and A Shaw in the sixties. This area became known as syntactical pattern recognition when formal language theory was used as a way to integrate symbols and their inter-relationships, the analysis of samples and the inference of grammars and languages from a set of symbolic samples. King-Sun Fu and his students produced most of the significant papers in this area from the early 1970's to his ultimately death in 1985 [8].

Formal language appear to be natural way to deal with the problems involved with symbolic features. However, it was soon realized that when the main issue is to emphasize feature analysis and a description of their relationships, then symbolic analysis could be treated by means of other mathematical tools, such as string, tree or graph matching.

Many pattern recognition problems do adopt the decision theoretic approach. The approach is to first extract a set of features that characterize the objects, and then represent them as points in R^n

* 계원조형예술대학, 정보통신과 교수 (sdkim@ns.kaywon.ac.kr) ** 광운대학교 컴퓨터공학과 교수
논문번호 : 00327-0816, 접수일자 : 2000년 8월 16일

for classification. The idea is that objects belonging to the same class will form clusters in $\mathbb{R}^n$. The recognition or classification process is now reduced to the problem of partitioning the feature space into regions where they correspond to the different classes of objects.

There are applications, however, where the structure of a pattern plays an important role in the classification process. In these situations, the decision's theoretical approach has serious drawbacks because it lacks a suitable formal structure for handling pattern structures and their relationships. [1]

The syntactic or structural approach is based on a very powerful idea that agrees with human intuition. The idea breaks down an object into a set of simpler sub-objects or sub-parts and each sub-part is in term decomposed into a set of even simpler sub-sub-parts, etc., until the simplest sub-parts are reached, in a hierarchical manner. The basic idea was that a complex pattern could be described recursively in terms of simpler patterns.[2] For example, an image may be decomposed into objects, the objects into parts, the parts into strokes, and the strokes into pixels[3]

The pictorial patterns shown in Figure 1.1 can be described in terms of the hierarchical structures shown in Figure 1.2. This approach draws an analogy between the (hierarchical, or treelike) structure of patterns and the syntax of languages. Patterns are specified as being built up out of sub-patterns in various ways of composition, just as phrases and sentences are built up by concatenating words, and words are built up by concatenating characters.

In order to apply extended formal language theory to an object or pictorial pattern recognition, first, the object has to be represented. The most common representation scheme is strings of symbols. Two other major representation are arrays[4] and graph representations[5]. A class of objects is therefore described by a language, also known as pattern description language[3] or pictorial description language[5], which is a set of symbol strings. The recognition process is to decide whether a particular string of symbols belongs to the class of objects. This is done using a grammar to decide whether the symbol is within set of rules suggested by a grammatical inference engine, which uses the training set of symbol strings from the class of objects under study.

The proposal idea of the structural or syntactical approach with regular shape model is to represent the pictorial pattern or objects in terms of symbols, strings, arrays, or graphs and then use it for classification using the theory of formal language with rules for a picture grammar. We, here, will show the result of implementing some examples to illustrate the idea.

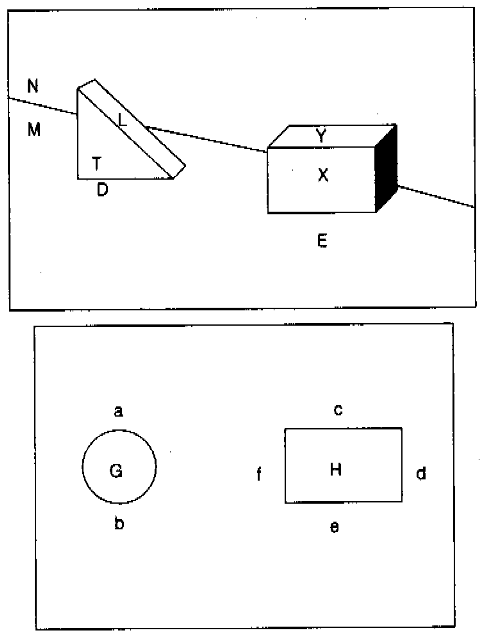
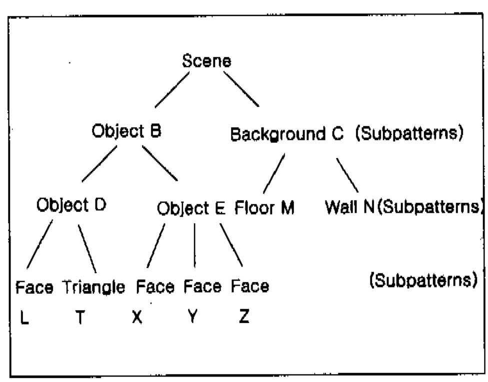


Fig 1.1 The pictorial patterns (a) and (b)



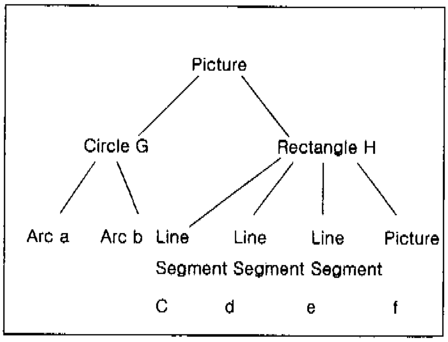


Fig 1.2 Hierarchical structure description of above (a) and (b)

Ⅱ. Simple shape analysis

2.1 Representing the boundary of objects

We consider the extremely simple case of a segmented silhouette, a rectangle, in a binary image as shown in Figure 2.1(a). To find the boundary of the rectangle, we merely remove its interior pixels. The resulting image then contains only the boundary of the rectangle, Figure 2.1(b). To represent the boundary of the rectangle, a common scheme is a chain code together with a starting position [8]. The starting position is P_0 and the chain code direction is indicated by the arrows, Figure 2.1 (c). The algorithm to find the chain code is the border-tracing algorithm. A simple descriptor of the chain code representation is the length of the chain code, and it is defined as the number of vertical and horizontal components plus $\sqrt{2}$ times the number of diagonal components gives the exact length[7], Figure 2.1(d). Finally, the descriptor may be used for classification. The above is one scheme using a continuous representation. The term continuous is used despite the fact that all values will be of the form $\alpha+\beta\sqrt{2}$, for positive integers α, and β.

An alternative approach which is intrinsically discrete is as follows: for example, line segments: horizontal line a, and vertical line-b, are used to represent the rectangle Figure 2.2(a). One of the descriptors based on this representation scheme is the discrete symbol strings: abab, Figure 2.2(b).

The classification method employed is grammatical string matching. This is generally associated with the syntactic approach.

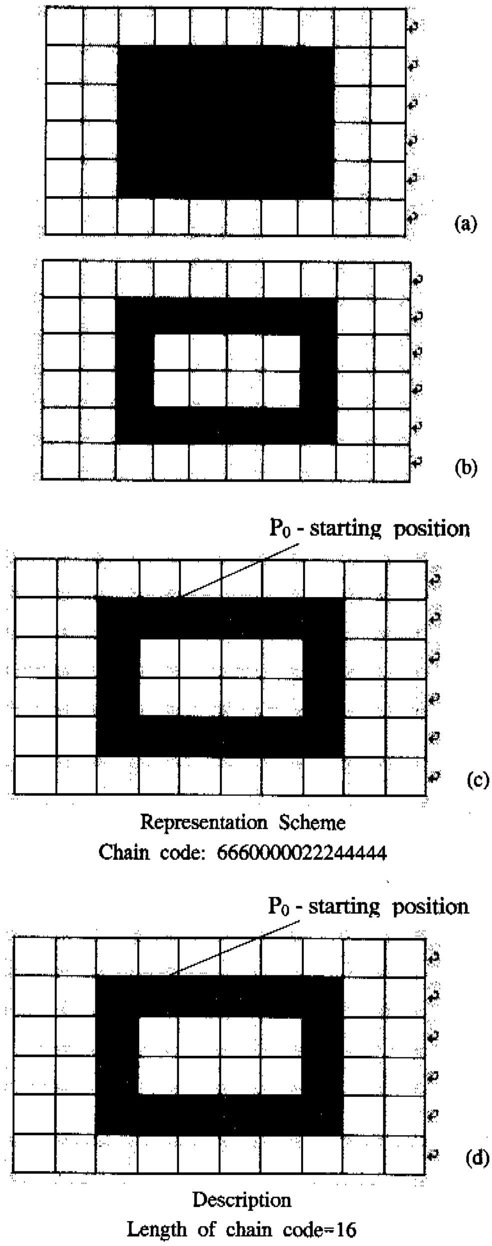
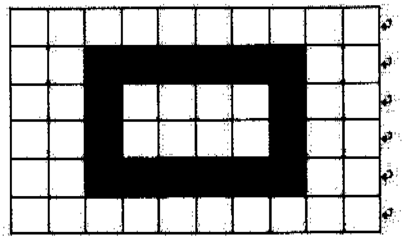
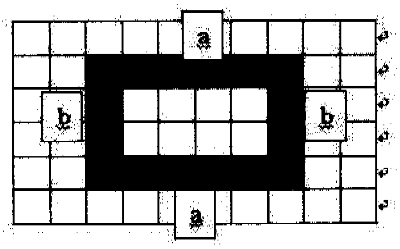


Fig 2.1 A continuous representation (a) the silhouette of the rectangle (b) the boundary of the rectangle is obtained simply by existing the interior pixels (c) the boundary of the rectangle is represented by the chain code and its starting position, P_0. (d) The descriptor of the boundary is the length of the chain code.



Representation Scheme

(a)



Description

(b)

Fig 2.2 the syntactic approach (a) the boundary of the rectangle is represented by the horizontal and vertical line segments a and b. (b) the descriptor of boundary is the discrete string symbol : *abab*

2.2 Selection of primitives

The first step in the syntactic recognition approach is the selection of the primitives. The choice of primitives and their relationships are represented by composite operations which determines the practicality of such an approach. Traditionally, the selection of a set of primitives is greatly influenced by the nature of the data, the specific application and the technology available. The following requirements usually serve as a guide for selecting pattern primitives[3].

- The primitives should serve as basic pattern elements to provide a compact but adequate description of the data in terms of the specified structural relations (e.g.,: the concatenation relation)
- The primitives should be easily extracted or recognized by existing non-syntactic methods since they are considered to be simple and compact patterns and their structural information is not important.

In general, there is no universal solution to the selection of primitives. This is similar to the situation of feature extraction problem in the Vector Space Representation approach.

Example 1) Given a thin-line representation of a triangle shown in Figure 2.3 (a) and the primitives shown in Figure 2.3 (b). The triangle is represented by the string *abc*, for example using the "concatenatation " operation from [3] Figure 2.3 (c). Different sizes of the triangle are therefore represented by the string a^m b^m where m 1, an example of a size 3 triangle is shown in Figure 2.3 (d)

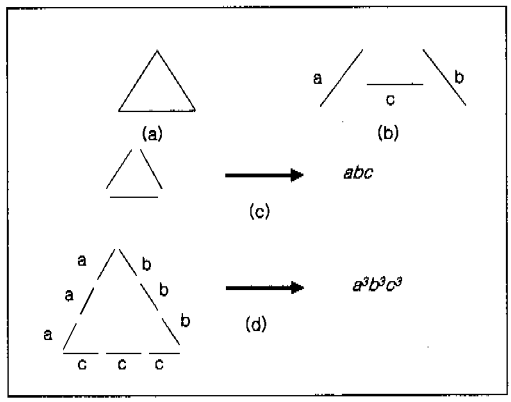


Fig 2.3 A simple example to illustrate the idea of structure
(a) A thin-line representation of the triangle
(b) Primitives and their discrete symbol representation
(c) Approximation of the triangle using the and symbol strings
(d) Approximation of a different size, 3, triangle and its symbol strings

The language *L(G)*, for the triangle is

$$L(G) = \{a^m b^m c^m : m \geq 1\}$$

And it can be generated by the following grammar,[6]

$$G = \langle \{S, X, Y\}, \{a, b, c\}, S, P \rangle$$

where P consists of the following production rules :

$$S \longrightarrow aSXY$$
$$S \longrightarrow aXY$$
$$YX \longrightarrow XY$$

$$AX \longrightarrow ab$$
$$BX \longrightarrow bb$$
$$BY \longrightarrow bc$$
$$CY \longrightarrow cc$$

The derivation of 'parse' of the triangle in Figure 2.3 (d) is as follows :

$S \Rightarrow aSXY$
$\Rightarrow aaSXYXY$
$\Rightarrow aasXXYY$
$\Rightarrow aaaXYXXYY$
$\Rightarrow aaaXXYXYY$
$\Rightarrow aaaXXXYYY$
$\Rightarrow aaabXXYYY$
$\Rightarrow aaabbXYYY$
$\Rightarrow aaabbbYYY$
$\Rightarrow aaabbbcYY$
$\Rightarrow aaabbccY$
$\Rightarrow aaabbbccc$

One of the main strengths of the structural or syntactic approach is the ability to represent a higher-level structure simply by building up from simple structures.

Example 2) Given the primitives shown in Figure 2.4 (a)[1], a higher-level structure can be gradually built up as in Figure 2.4 (b) to (g). The symbol~indicates a reversal in direction of primitives. The pattern in Figure 2.4 (g) is a composition of the patterns in Figure 2.4 (d) and (f).

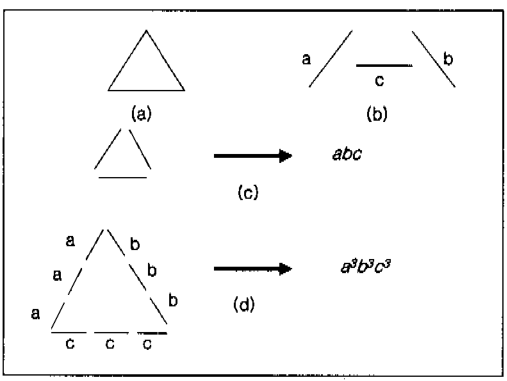


Fig. 2.4 Example of gradually extracting higher-level structure (a) Primitives (b)-(g) Representation of some simple pattern

Ⅲ. Formal language theory

The following terminology can be found from the following texts about grammars and languages [3][6].

Let γ be the alphabet, a non-empty set. The elements of γ are called the symbols, or letters, or characters. A string or word is a finite sequence of symbols. γ^* is defined as the set of all finite length strings of symbols of γ, including the empty string, ε ; $\gamma^+ = \gamma^* - \varepsilon$. The language L is generated from a device called a generative grammar or rewrite grammar G. A rewritten grammar consists of a finite set of rules or production which specify the syntax of the language. |x| is defined as the length of the string x, i.e, the number of symbols composing the string. Let x∈,the x^n is xxxx……. written n times.

The language generated by G, $L(G)$, is the set of strings in the sentential forms that contain only the terminal symbols:

$$L(G) = \{\eta \in V_T^* : S \xrightarrow[G]{} \eta\}$$

$S \rightarrow x$, is used if it is clear which G is involved, means that the terminal string x is derived from the starting symbols S using grammar G.

Ⅳ. Classification

Given m classes of patterns, G_i, i=1......m and the m grammars, G_j, j=1,.......,m, inferred, for a new input pattern string, x, the problem is to decide whether $x \in L(G_j)$ for j=1,......m ? If x belongs to $L(G_i)$, then it also belongs to class i. If x belongs to more than one language, then no decision is made because it is ambiguous. Finally, x is rejected if it does not belong to $L(G_j)$.

One of the simplest implementation is to match the input pattern string, x, with the prototypes from each class. The input pattern x will be arranged the class of the best match based on some similarity criteria. This approach is only useful when the appropriate prototypes and similarity criteria can be determined.

A finite state grammar, G_{i},, uses a deterministic finite-state automaton to recognize the string generated by G_{i},, whereas a context-free or a context-sensitive grammar requires a non-deterministic automaton. The process of performing the recognition is called 'syntax analysis', which ,in general, uses a non-deterministic procedure. The outcome of this process not only produces the decision of the input pattern *x*, but also the derivation, or parse of the string in a tree structure, which shows the decomposition of the pattern in terms of the sub patterns.

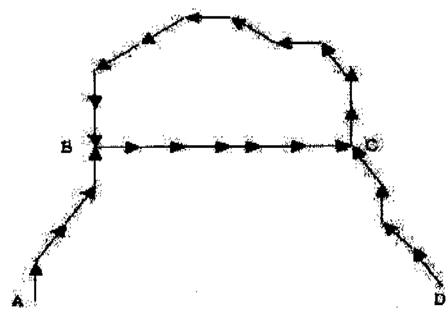
Consider the grammar $G= (V, V, P,S)$ where $V_N= \{ S,T,I \}$, $V_T= \{a,b,c, +, *\}$
$I \in \{a,b,c\}$ and *P*:

$$S \Rightarrow T$$
$$S \Rightarrow S+T$$
$$T \Rightarrow T*I$$
$$T \Rightarrow I$$

The derivation tree of the sentence $x= a*b + c*a + b$ can get easily.

Example 3) To deal with patterns of irregular shape, more sophisticated methods are employed. To describe the structural relationships within the patterns. Knoke and Willy applied a "transformation grammar "on the chain code to first reduce the irregularities of the pattern and then substitute the chain-coded string by some arc-shape labels. For example [3], the initial description of the character A is shown in Figure 4.1. The character is represented by four segments where each segment is represented by a chain code string.

Using the transformation rules, we substitute the chaincode string description by arc-shape labels. Each pattern is now represented by a finite, planar, directed, connected graph, and the structure of such graph can be described in terms of its arcs and vertices. Since the number of the



Chain-encoded Arc-shape encoding	Relative direction	Initial vertex	Ter. Ver.
0110	0	A	B
222222	2	B	C
0076765544	6	C	B
7707	7	D	C

Fig. 4.1 An initial description of the character A

arc-shape labels and the number of the vertices are finite, the language L_c used to describe the character is a finite-state language.

A four step matching algorithm is used in the recognition process to match a new character description to a directory of L_c prototypes. The matching algorithm is as follows:

Step 1: Obtain a standard form description for the "unknown" input pattern.

Step 2: Property screening Check the number of vertices and the arc-shape labels as a preliminary screening.

Step 3: Structure matching ignoring relative directions-Create all possible relabelings of prototype vertices to attempt to obtain a restricted (ignoring relative directions) one-to-one match between each prototype description and the "unknown" description.

Step 4: Complete structure matching A candidate prototype which results from the step 3 matching must be checked exhaustively for relative direction matches.

If a match is found, recognition has been completed. If not, the algorithm returns to Step 3 unless all possible vertex relabelings have been generated and checked.

4.1 Simulation of proposed approach

The system consists of three of major modules, selection of primitives, grammatical inference for regular, syntactical approach classification, and output, as shown in Figure 4.1. To illustrate the procedures required at each composition process, let us consider the rectangles, triangles and rhombuses. It is clear that each of them requires two composition processes.

1. from pixels to line segments (set of primitives), for the first process;
2. from line segment to objects(rectangles, triangles, rhombuses), for the second processes.

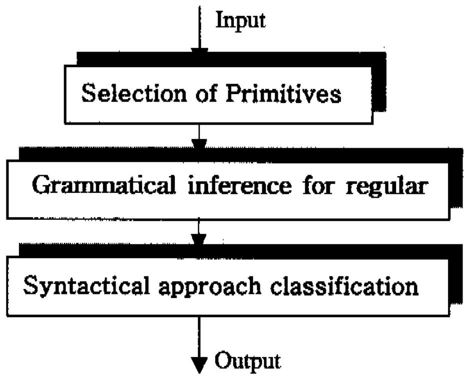


Fig 4.2 Information processing module

In every composition process, there is a set of sub-parts that form a part. For example, in the first stage, the sub-parts are the pixels and the parts are the line segments. In the second process, the sub-parts are the line segments and the parts are shapes, which are the objects, finally we will get the formal languages. Primitives are usually the simplest parts of the hierarchy, in this case, they are the line segments. To avoid confusion, if there exists a name for the sub-parts and the parts, as in this case, they will be used. Otherwise, the generic terms sub-parts and parts will be used.

Therefore the procedures given for simulation results, Figure 4.3 to Figure 4.5, are

1. find the line segments in the first process;
2. find a description of the line segments;
3. find the triangles, rectangles, rhombuses in the second process;
4. find the descriptions

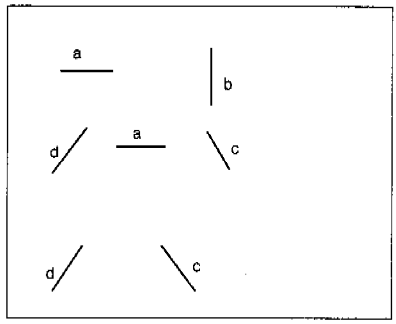


Fig 4.3 Initial primitive sets for regular shape

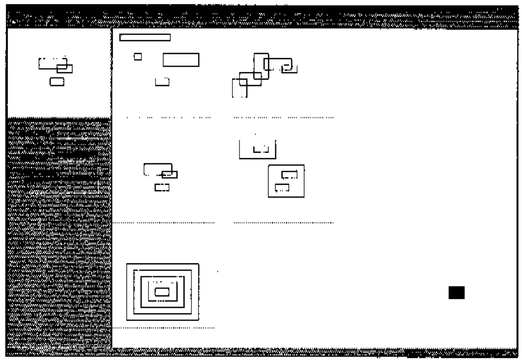


Fig 4.3 Recognized strings of rectangle regular shapes

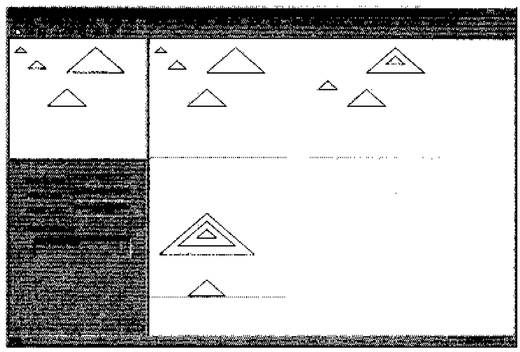


Fig 4.4 Recognized strings of triangle regular shape

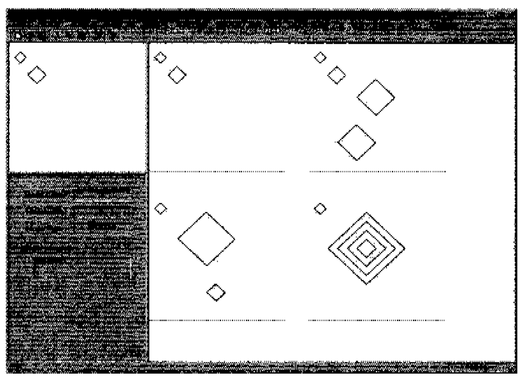


Fig 4.5 Recognized strings of rhombus regular shapes

V. Discussion and Conclusion

We proposed that the idea of decomposing an object into sub-object, etc., is a powerful idea, but the representation of the primitives by symbols and the description of the patterns by strings of symbols is not readily implemented. The fact that it is not possible to infer a pattern grammar, which will generate only the valid patterns, has obstructed the progress of this approach, and the choice of primitives which forces the geometric objects into algebraic objects is ad hoc and imposes premature quantization of the objects.

Some attempts have been made to deal with the problem of forcing the geometric objects into algebraic objects by putting a metric on the resulting strings, but this has not been as successful as might be hoped. Nishida has attempted to preserve the geometry by specifying line segments by giving the coordinates of their endpoints and to preserve the structure by classifying the types of intersection of line segments.

This paper has described the fundamental issues involved in building a VOSR (Visual of Shape Recognition) system from the treatment of the image to the classification by decision theoretic and structural methods. The attractions of the structural methods are great, but there are great difficulties in implementing them.

References

[1] Gonzalez, Rafael C and Thomason, Michael G, Syntactic Pattern Recognition-An Introduction, Addison-Wesley, 1978

[2] Pavlidis, T, Structural Pattern Recognition, Springer Series in Electrophyisic I,Springer-Verlag, 1977

[3] Fu, King Sun, Syntactic Methods in Pattern Recognition, Academic Press, 1974

[4] Wang, P.S , Array Grammars, Patterns and Recognition, Vol 18 of series in Computer Sciences, World Scientific, 1989

[5] Shaw, A.C, A formal picture description scheme as a basic for picture processing system. Information and control 14:9-52, 1969

[6] Tremblay, J.P and Manchar, R, Discrete Mathematical structures with Application to computer sciences, McGraw-Hill, 1975

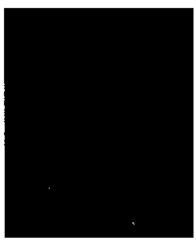
[7] Gonzalez, Rafael C and Richard E, Woods, Digital Image Processing, Addison-Wesley Publishing Company, Inc, 1992.

[8] Yuan, Jianxing and Suen, Ching Y, Generalized chain codes based straight line detection, World Scientific, 1992

[9] D.Dori, and A. Bruckstein, Shape, structural and Pattern Recognition, World Scientific, ISBN 981-02-2239-4, 1995

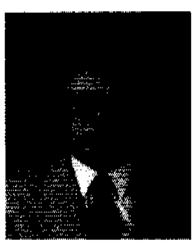
[10] Nishida, H, Automatic construction of structural models incorporating discontinuous transformations, IEEE Transactions on pattern Recognition and Machine Intelligence, Vol. 18, no 5, pp 400-441, April, 1996

김 성 동(Seong-Dong Kim)



1981년 : 광운대학교 응용전자과 졸업
1984년 : 아주대학교 대학원 전자공학 석사
1991년 : 뉴욕시립대학교 대학원 전산과 석사
1994년 : 미국 스티븐슨 공대 박사수료
1998년 : 광운대학교 컴퓨터공학과 박사수료
1995년~현재 : 계원조형예술대학 정보통신과 교수
<주관심 분야> Object recognition and tracking, digital image processing Structural and Syntactic pattern recognition

최 기 호(Ki-Ho Choi)



1973 : 한양대학교 전자공학과 졸업
1977 : 한양대학교 대학원 전자공학과 석사
1987 : 한양대학교 대학원 전자공학과 박사

1977～1979 : 한국과학기술원 연구소 연구원
1989～1990 : University of Michigan Visiting Scholar
1979～현재 : 광운대학교 교수
<주관심 분야> 멀티미디어 정보검색, Object recognition and tracking