tanulmányok

9/1973

MTA Számítástechnikai és Automatizálási Kutató Intézet Budapest

MAGYAR TUDOMÁNYOS AKADÉMIA
SZÁMITÁSTECHNIKAI ÉS AUTOMATIZÁLÁSI KUTATÓ INTÉZETE

PICTURE PROCESSING USING PAX

R.Narasimhan
Tata Institute of Fundamental Researc
Bombay 400005

A tanulmány a szerzőnek 1973. májusában Intézetünkben tartott előadását tartalmazza

1973

A kiadásért felelős:

Dr. Vámos Tibor

az

MTA Számitástechnikai és Automatizálási
Kutató Intézet

igazgatója

736245 MTA KESZ Sokszorosító. F. v.: Szabó Gyula

PICTURE PROCESSING USING PAX

R.Narasimhan.

Tata Institute of Fundamental Research

Bombay 400005.

NOTE: The notes that follow are extremely brief and are certainly not intended to be an introduction to PAX in any of its several versions. PAX II is discussed adequately in [1] and COMPAX in [2], [3] contains also an account of the parallel processing framework of which PAX is a machine implementation. Rosenfeld's book [4] summarizes usefully most of the picture processing techniques but does not relate them specifically to PAX. My intention here is to outline the characteristic strong points of PAX for picture processing and consider those techniques most suitable for use with PAX.

R.Narasimhan
Rome. June 15, 1973.

1. Emily G. Johnston: The PAX II Picture Processing System; in BS Lipkin & A Rosenfeld (Eds): Picture Processing & Psychopictorics, Academic Press, 1970; pp. 427-512.

2. R.Narasimhan: Syntax-directed Interpretation of Classes of Pictures; Comm. ACM, 9 (1966), 166-173.

3. R.Narasimhan: Labeling Schemata & Syntactic Descriptions of Pictures; Info. & Control, 7 (1964), 151-179.

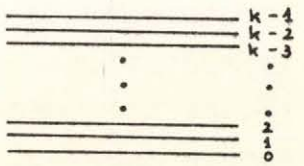
4. A.Rosenfeld: Picture Processing by Computer, Academic Press, 1969.

1. Where PAX is most advantageous to use.

1.1 Processing of pictorial information using digital computers is perhaps most readily summarized in terms of the following table. Here, the categorization has been done in terms of the kinds of pictorial inputs that have been used, the various processing techniques that have been developed, and the several motivations for processing pictures.

(A) WHAT (Nature of the Input)	(B) HOW (Nature of the processing techniques)	(C) WHY (Nature of the motivations)
(1) Images digitized from photographs, slides, etc. a) black-white b) gray levels c) color (2) Line Drawings a) Symbols/ characters b) Sketches c) Biomedical data e.g. EEG,ECG, etc.	(1) Global Transformations a) Fourier b) Hadamard c) Freeman Encoding d) Curve Fitting (2) Preprocessing using arithmetic & Logical Transformations Homogeneously a) noise cleaning b) noise addition c) smoothing(averaging)	(1) Encoding for combating noise, bandwidths compression, etc. (Signal Processing) (2) Transforming for assistance to subsequent human processing

A multivalued picture (ie., several gray levels) is stored in a stack of PAX planes. The most natural extension (and one that makes it possible to carry out certain kinds of preprocessing operations very easily) is to use a separate plane for each gray level. Thus, a k-valued gray-scale picture will be stored in a stack consisting of k PAX planes, with values 0 upto k-1 in ascending order, as shown.



A stack of k-planes to store a k-valued picture

This may be considered to be somewhat wasteful of storage space in an actual machine implementation. PAX II uses binary-encoding. Here a stack of planes is used so that the bottom-most plane refers to gray value 2^0, the next one to gray value 2^1, the next to 2^2, the next to 2^3, and so on. Thus to store a 32-level picture only 5 planes are needed, and not 32 as in the earlier case. As we shall see later, binary encoding enables arithmetic operations to be carried out more readily.

Primitive operations in PAX are functions whose arguments are one, two, or three PAX planes. The intermediate and final results are written in PAX planes again.

A PAX plane, thus, plays the role of a (linear) register in conventional arithmetic computers. A plane is a natural extension to 2-dimensions of the string concept. Every symbol in a string is characterized uniquely by its position starting from the first (or last) symbol

not be completely true of Freeman encoding procedures). A similar criticism applies to the property-vector classificatory approaches (Technique 3) when applied in a local sense.

For objectives (1) and (6), for the application of Techniques (1) and (3) and also for input category 2(c), pictures are best thought of as numerical matrices. "Picture" processing is actually converted into the processing of numerical matrices, i.e., to arithmetic computations. The use of PAX has no special advantage in these circumstances. In fact, PAX is very inefficient where large-scale arithmetic processing is required.

But PAX has been found to be extremely powerful and efficient where picture processing is carried out within the pictorial domain; that is, where the interpretation assignment process remains uniformly valid throughout all the intermediate stages of processing a given input picture. Such a situation naturally arises in the case of picture categories 1, 2 and 3, processing techniques 2 and 4, to meet the objectives 2,3,4,5 and 7, especially.

2. The PAX Mode of Operation

2.1 The fundamental aspect of PAX that gives it its power in pictorial operations is its data-structure. The primitive data unit is a PAX PLANE consisting of an nxm raster of binary-valued points: (In PAX II, n and m can be programmer defined; it is usual to operate with square rasters by setting n=m). A two-valued picture (black-white) is thus stored in a single PAX plane as a retinal image.

1.2 Not all techniques are useful and/or relevant for all objectives, or with all the different kinds of inputs. Global transformations have been mostly used with digitized photographs (satellite pictures, cloud-cover pictures, biological images, etc) especially in the case of objectives (1) and (2), Many of the preprocessing techniques (especially the arithmetic ones) have also been developed and used in this context.

But the relevance and/or usefulness of global transformations (especially the orthogonal transformations) in the context of object articulation and scene analysis (objectives 4,5,7, especially) is not at all clear. For, to meet any of these objectives pictures have to be analyzed from the point of view of their representational aspects. That is, we assume explicitly that the input pictures relate to well-defined extra-pictorial situations. We want to analyze the inputs in order to gather information about these situations. This is what is meant by saying that one is conceined with <u>interpreting</u> (i.e., assigning interpretations or readings to) the input pictures. Interpretation of any complex situation necessarily presupposes the articulation of aspects of this situation. That is, we want to isolate the component parts of the situation and exhibit the interrelationships between these component parts. In this sense, any interpretation of nontrivial situations calls for the use of some kind of segmentation procedure. The principal drawback of the global transformation techniques consists precisely in the fact that they do not allow us to segment a picture into components that are meaningful from the interpretational point of view: (this, however, may

(A) WHAT (Nature of the Input)	(B) HOW (Nature of the processing techniques)	(C) WHY (Nature of the motivations)
d) Engineering Drawings e) Scientific Research Data. e.g. Bubble-chamber Pictures, etc. (3) 3-D Scenes e.g. TV Camera input	d) sharpening (differentiating) e) shrinking/ thinning etc. (3) Classification based on Property Vectors (4) Scene Analysis a) object articulation (i) Figure-Ground Separation (Region Analysis, Segmentation, etc) (ii) Feature Extraction (b) Relational Structure Articulation.	(3) Automatic Discrete Symbol Recognition (4) Automatic Biological Image Analysis and Recognition (5) Automatic Bio-medical Diagnostics (6) Computer-aided Engineering Design. (7) Simulating (modelling) animal visual perception.

TABLE 1.

of that string. In the case of a plane, analogously, we need two indices to characterize a point (for example, its x-y coordinates with reference to some fixed coordinate system).

Let us denote a generic point in a PAX plane by (i,j). Unlike as in the case of a string where each point (i) has only two immediate neighbours (i-1, i+1), each planar point (i,j) has eight immediate neighbours; (a rectangular grid is used in PAX). For ease of reference, the individual neighbours are identified by direction numbers as shown.

4	3	2
5	(i,j)	1
6	7	8

Direction Numbers identifying the immediate neighbours of (i,j).

Larger neighbourhoods can be defined as required for specific tasks by extending the direction numbers to the next level of neighbours. Special neighbourhoods can be selected by restricting processing to a subset of direction numbers. For example, the list (1, 3, 5, 7) restricts the neighbourhood to the East-West-Norths-South neighbours only.

2.3 The primitive operations of PAX can be grouped into the following categories.

1. Single Plane Operations.

a) Read into a Plane. (Input operation).

b) Read from a Plane, ie., Print out the raster of points (Output Operation).

[Note that Input and Output could refer to a stack of planes in the case of multiple level pictures. The output can also be printed out in

gray scales through suitable use of symbols and/or overprinting of symbols.]

(c) Write 1 in point $P_1 \equiv (x_1, y_1)$
[Note that arbitrary masks or context planes can be generated through repeated use of this instruction.]

(d) List the set of points in the Plane (i.e. list their (x,y) coordinates).

(e) Clear the plane.

(f) Complement the content of the plane.

(g) Shift the contents one step in the $\pm$X direction; $\pm$Y direction.

(h) Mark specified neighbours of (i,j) (for all (i,j) in the Plane).

(i) Picture transformations based on Boolean functions of specified neighbours of (i,j).

(j) Thresholding operation: A neighbourhood for point (i,j) is specified by a list of direction numbers. If the number of neighbours of (i,j) exceeds a given threshold T, then (i,j) is marked in the output plane; otherwise, not.

2. Two Plane Operations

(a) Transfer between planes.

(b) AND, OR of contents of two planes.

3. Operations with a Context Plane

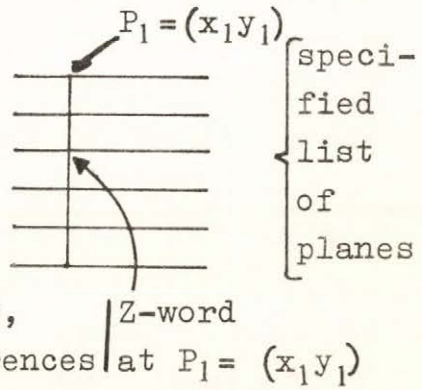
(a) All operations of (1) and (2) with another specified plane used as a context plane, i.e., as a mask: only the subset corresponding to the points in the mask enters into the computation.

4. Read Z-word

(a) Read Z-word (made up of the bits in a specified list of planes) corresponding to the point P_1 (x_1, y_1). (The bits in the Z-word correspond to the (X_1, Y_1) point in the specified list of planes). See figure.

5. Other operations

The operations listed above are the most primitive picture operations. To construct programs within PAX, other sequencing operations such as jumps, tests (test if a plane is zero), indexing of arguments, and so on are needed. The references given in the preface should be consulted for more details.



2.4 We shall see later on how, using these primitives, more complex operations can be programmed: for example, connectivity operations. We shall review in the next section some of the more important picture processing techniques that have been developed in B2 and B4 (see Table in Sec. 1) and consider the advantages of PAX in implementing these techniques.

But before doing this it is important to introduce the concept of labelling in the context of the PAX mode of operation. A label, primitively, can be thought of as associated with each PAX plane that contains a picture

or part of a picture (as a result of processing). To begin with, the gray scales themselves are labels. Every step in the processing results in one or more pictures which can be thought of as being identified by specific labels. Examples of such labels are: figure, ground, interior points, border points, junctions, corners, bends, end points, and so on. At a higher level, one could have labels naming object parts, objects, subsets of scenes, and so on. These labelled planes, clearly, could be used as context planes for specific intermediate level processing; for example, those corners interior to this island; corners connected to these end points, all objects O, in the top-left region of the scene, and so forth. Picture processing can, in fact, be thought of as essentially concerned with label generation. Our subsequent discussions will be precisely concerned with the classes of labels that have been found to be useful and ways of generating them.

The Z-words can be probed to see (in terms of their patterns) what subsets of labels particular subparts of scenes have generated. Z-words, in effect, are feature-vectors. Evidently, scene segmentation, as also object articulation, can be effected very efficiently using the Z-word patterns.

3. Review of Some Preprocessing Techniques Using Local Transformations.

3.1 In this section we shall review some of the more important algorithms that have been developed to preprocess

pictures using local transformations homogeneously. We shall specifically be interested in noting the effectiveness with which these operations could be carried out using PAX. This would enable us to appreciate not only the picture processing power of PAX, but at the same time the kinds of labels that can be generated and how these labelled planes can be effectively made use of in sequential processing of pictures through several stages.

3.2 Approaching picture processing from the view-point of structure analysis (i.e. analysis keeping explicitly in mind the underlying picture syntax), three distinct stages can be distinguished. These are:

(1) Preprocessing
   a) noise cleaning
   b) figure-ground separation
   c) image idealization and/or refinement.

(2) Single object feature analysis and abstraction.

(3) Multiple-object relational structure analysis (Scene Analysis).

Notice that although logically stage (1) precedes stage (2), and stage (2) precedes stage (3), it does not follow that in practice it is either efficient or even possible to carry out the analysis of a given picture sequentially, stage by stage, and process each stage in isolation from the others. This may indeed be possible in the case of very simple pictures, i.e., pictures with very simple semantics. But even with moderately complex pictures, e.g., handprinted alphanumeric

characters, it becomes essential to carry out the processing in a form that is not strictly hierarchic. It has now become fashionable to use the term 'hetararchic' to denote this kind of processing, to contrast it with strictly hierarchic processing schemes. Hetararchic processing typically involves the following:

(a) Partially process the ith stage making use of the partially processed outputs from the first up to the $(i-1)$st stages. Notice that some of these outputs could be zero since the processing of those stages may have been skipped.

(b) In the case of ambiguities and/or other local problems, go back to the j th stage $(j < i)$ for more complete processing to resolve the current problems satisfactorily. Return to the i th stage and repeat (b) if necessary.

(c) Proceed to the k th stage $(k > i)$ with the partially processed outputs of stages 1 up to i. This is the same as (a) with k replacing i.

To carry out efficiently hetararchic processing of this type two kinds of facilities are essential. The first is an appropriate data-structure. The second is powerful problem solving heuristics to make the relevant decisions at the various stages of processing. Our concern here is restricted to considerations of PAX from the viewpoint of the data structure that it makes available. It must be noted that the PAX data structure need not be applicable withs uniform efficiency at all stages. In fact at the higher stages when the processing decisions have to be based on global

relational considerations, the inputs to these decisions may very well be string language expressions, e.g., natural language statements, or statements in some specially constructed formal language.

We shall argue the thesis that, in spite of the limitations noted above, processing within PAX makes available partially processed labelled outputs which could be very powerfully used during all the three stages of processing earlier indicated. In this section we shall concentrate on the preprocessing stage.

3.3 Noise cleaning

Let us consider, to begin with, binary pictures and assume that black denotes 1 and white, 0, and that black points constitute the figure.

(1) Elementary noise cleaning procedures are based on removing isolated black points and filling-in isolated point-gaps.

At a somewhat more sophisticated level one could base point removal and point addition on thresholding: add (i,j) if 5 or more neighbours are present; remove (i,j) if 3 or less neighbours are present. Edgesmoothing can be achieved to a certain extent by this procedure. But care should be taken to ensure that these operations when iterated do not end up with a completely black or completely white picture.

The above procedures deal with the entire picture uniformly, by performing this smoothing operation on subpictures with specific labels, edge-smoothing and so on can be restricted to specific parts of the picture: for example, diagonal line element edges can be smoothed preferentially without affecting the thickness of N-S or E-W lines; Nearly circular blobs can be smoothed without affecting thin, long, ellipsoidal blobs; and so on.

(2) More general smoothing procedures have been effectively used and are applicable to gray level pictures. The following list is taken from Rosenfeld (see Preface for reference).

(a) Replace each gray level by the average over a specified neighbourhood of that point. Weighted averages can be used, the weights going down for points farther away from the centre (i,j).

(b) Smoothing, in general, blurs a picture. To achieve smoothing without blurring:

(i) average a set of independent copies of a picture pointwise. Assuming noise is uncorrelated between pictures, the picture points preserve their value, but noise points are smoothed out.

(ii) replace gray level of (i,j) by the average gray level over a specified neighbourhood provided this average is above some specified threshold. Otherwise leave (i,j) unaltered. This process does not blur the picture but tends to smooth out isolated spots and streaks.

Notice that smoothing operations of the type listed under (i) can be immediately performed within PAX. Operations listed under (2) require arithmetic computations and these can be carried out within PAX only by indirect methods. PAX II uses logical procedures to perform additions. A second method is to use the "Read Z" instruction point-wise. But this is very laborious.

As was pointed out earlier, PAX is not an efficient language in which arithmetic involving the values of neighbourhood points can be readily carried out. It would seem to be plausible to argue that perceptually relevant picture processing should not require the computational procedures of exact arithmetic, like summing, multiplication, division, etc., except in the very elementary sense of counting and thresholding. It is an instructive research program to study to what extent preprocessing operations that have been extensively utilized and which are based on exact arithmetic computations could be replaced by processing operations based only on counting and thresholding. The replacement, in general, is not likely to generate identical results. But what one is asking is whether replacements could be found which give rise to acceptable and satisfactory (adequate) alternatives. If such replacements are available, they should be preferred in so far as they can be readily implemented within PAX and lead to significantly more efficient processing.

As an illustration, in the multigray level case, instead of computing arithmetic averages and thresholding the results, it seems likely the following alternative smdothing procedure would give equally acceptable results:

Let N be the total number of points in the neighbourhood of (i,j) ; in the 3x3 neighbourhood, N=8 (excluding the point (i,j) itself). If at least $K \leqq N$ of the neighbours have gray values $\geqq G$, then replace the value of (i,j) by the largest, maximally occurring neighbouring value. Else, leave (i,j) unchanged.

Exercise: Consider a picture with 8 gray levels: 0,1,...,7. Here zero represents the ground. Assume that the figure is completely surrounded by a border of zeros so that shifting does not lose information. Use 3x3 neighbourhood (N=8). Take K=G=5. Write a flow chart for a PAX program to implement the above smoothing process.

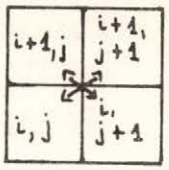
3.4 Figure Abstraction, Refinement & Segmentation

1 Edge enhancement or Sharpening

Smoothing, as we saw, is based on averaging or integration and tends to blur a picture. The inverse operation of differentiation enables one to generate a transform of a picture that tends to emphasize the boundaries between uniformly gray regions. The usual procedure that is adopted consists in replacing the value at each point by the value of the gradient at that point.

The gradient is determined by computing the first differences in two orthogonal directions. For example, the gradient $\nabla(i,j)$ at $(i\ j)$ can be computed as:

$$[\nabla(i,j)]^2 = [(i,j) - (i+1,j+1)]^2 + [(i,j+1) - (i+1,j)]^2$$



For simplicity of computation, the above is usually replaced by the formula:

$$\nabla(i,j) = |(i,j) - (i+1,j+1)| + |(i+1,j) - (i,j+1)|$$

involving a sum of absolute values of the first differences.

Clearly, $\nabla(i,j)$ is zero inside regions of uniform gray level. To achieve some amount of noise cleaning, the gradient picture is usually thresholded. That is, for some specified t, only those points where the gradient value is $\geq t$ are retained in the output picture. If after thresholding, the output is generated as a binary picture, it can be more readily dealt with for subsequent shape analysis.

For better noise cleaning, sometimes the gradient picture is computed on the basis of averaged out first differences. For instance, the following weighted averaging has been suggested:

grad at e = $|Sx| + |Sy|$

where $Sx = (c+2f+i) - (a+2d+9)$

and $Sy = (a+2b+c) - (g+2h+i)$.

a	b	c
d	e	f
g	h	i

Computing the gradient picture in the above sense is, again, not easy to accomplish within PAX. But an acceptable differentiation or edge-generation procedure for perceptual purposes should be the following: given a picture with $(k\pm1)$ gray levels, $0,1,2,\ldots,k$, first of all smooth the picture by merging neighbouring gray levels to form a picture with r gray levels $(r < k)$. Now, remove all points with m or more (for some suitable m, say, m=6 for an 8-neighbour case) neighbours having the same gray level.

(2) Thinning

Some other picture refinement procedures are the following:

(1) shrinking a blob to a point (useful in blob counting). But what happens in the case of overlapping blobs? This is a nontrivial problem which can only be solved by context-dependent heuristics.

(2) thinning of line elements. This, in general requires directional labelling so that thinning can be confined to the 'width' of the line elements.

Exercise Devise on algorithm for thinning N-S and E-W lines in a picture to a thickness of one or two points. Hint: Define suitably interior points and boundary points and strip off boundary points only. Iterate till only boundary points are left. Ensure that connectivity is not lost. (See the section on "connectivity" further below.)

(3) Picture Segmentation

A picture is segmented by forming subsets of pictures according to various criteria. Notice that the most primitive segmentation is the figure-ground separation. Except in the case of simple pictures (e.g., single-object pictures, or where the gray scales of the background differ significantly from those of the figure) this is a non-trivial operation and may, in fact, constitute the most inportant part of perception. Notice also that the segmentation process merges with feature analysis at the higher end of picture abstraction. But certain types of segmentations can be thought of as truly primitive since they are picture-class independent: (the following is taken from Rosenfeld).

Given a picture f and an operation Φ that takes pictures into pictures, and a number t, we can segment f into a subset $f^{*}_{\Phi,t}$ by defining:

$f^{*}_{\Phi,t} \equiv$ the part of $\Phi(f)$ with gray level $\geq t$.

Consider as examples;

(1) Φ is the identity operator. One can take as 'figure' all subsets with gray levels $\geq$ t.

(2) Φ is a smoothing or noise cleaning operation, or a sequence of such operations. Now, apply thresholding to segment.

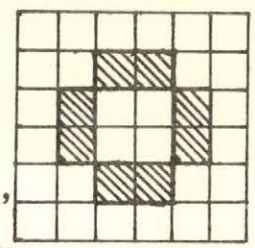
(3) Φ is a contouring operation (generates a gradient picture). Now apply thresholdig to segment outline figures.

One difficult problem is the choice of the threshold t. Sometimes the frequency distribution of the gray levels is investigated to choose t: (see Rosenfeld, pp. 129-133).

From a given subset obtained by thresholding other subsets can be generated by considering connectivity.

(4) Connectivity

Anomalous situations arise if the connectivity of a point (i,j) to its immediate neighbours is based on one definition of connectedness uniformly. This is readily seen from the accompanying figure which shows a simple closed curve in a plane. If (i,j) is assumed connected to any of its 8 immediate neighbours, provided both are parts of 'figure' or both are parts of 'ground', then the "inside" of the closed curve and its "outside" are connected, and the closed curve does not have an inside as a separated region. If, on the other hand, one uses 4-connectedness (E-W-N-S points only are connected to (i,j)) uniformly, the 'inside' and 'outside' of the figure are unconnected. But, equally, the sides of the figure themselves are unconnected. The figure is no more a closed curve but yet it has a distinct 'inside'!



One solution that has been suggested to get around this difficulty is to use 8-connectivity for figures, and 4-connectivity for ground, or vice-versa.

If one restricts the term 'neighbours' to connected neighbours, then interior and 'border' points of set S are readily defined. Similarly, it is easy to define an isolated point.

Points $a,b \in S$ are connected if there exist points $a=a_1,a_2,\ldots,a_{n-1}, a_n=b$, such that all $a_i \in S$, and a_i is a neighbour of a_{i-1} for $i=2,\ldots,n$.

A connected component of S is one each pair of whose points is connected.

Given a figure S, assume that S is surrounded by a border of zeros. The component of $\overline{S}$ consisting of elements connected to (any one of) these zeros is called the outside of S. If $\overline{S}$ has any other components, they are called holes in S. Any element of a hole is said to be inside S. If S is connected and has no holes, it is simply connected. Otherwise, it is multiply connected.

Notice that within PAX all the above connectivity-based labelling (inside, outside, interior points, border points, isolated points) can be immediately done. If figure S consists of

many disjoint components (islands), each can be separated by a labelling and subtraction (set subtraction) operation within PAX.

(5) Region Analysis

Region Analysis is a technique used for picture segmentation that is based on connectivity considerations. For the simplest case, each region is composed of picture points having the same gray value and connected to each other.

Call a set of points R in a picture an elementary connected region, provided,

(a) all points in R have the same gray value
(b) any two points in R are connected by a chain of adjacent points each of which is in R
(c) $R^* \supset R$ implies $R^* \equiv R$

That is, in the terminology introduced earlier, R is the largest connected component all of whose points have the same gray value.

Region analysis can be based on segmenting a picture into elementary connected regions. But, in general, smoothing operations are needed to avoid spurious regions being generated due to noise. One approach is to merge two adjacent regions if the average difference in the gray values of points on each side of a common boundary is not greater than some threshold t.

See: CR Brice & CL Fennema: Scene Analysis Using Regions; Artificial Intelligence, 1 (1970), 205-226.

It is clear Region Analysis can be carried out efficiently using PAX. As we pointed out earlier, smoothing operations within PAX are best performed when not based on exact arithmetic operations.

4. Feature Extraction and Single Object Shape Analysis

4.1 Line Pictures

Pictures made up of line-like elements are of fundamental importance. Even pictures composed of regions are often reduced to their outline drawings before detailed structure analysis. Such outline pictures are $\not\equiv$ again, clearly, line drawings.

Appropriate labels for line drawings are subsets of points which are of the nature of: (1) bends, (2) junctions, (3) corners, (4) terminals, etc. In general, to isolate such features, basic directional labels have to be assigned: N-S line, E-W line, Diagonal line, and so forth. In terms of cooccurences of these labels, junctions, etc., can be suitably defined.

At a more global level, line elements can be labelled as closed, open, curved, straight, etc. If some measure of curvature can be computed, one can use the labels, "gently curved", "tightly curved", and so on.

More elaborate processing is needed to classify line figures as: circle-like, oval-like, spiral-shaped, helical-shaped, and so on. (See below also under shape analysis).

4.2 Shape Description

Suppose we are given an arbitrary set of points in a plane forming a region of some sort, and we are asked to describe it. There would be little point in trying to describe it by enumerating all its points. We are interested in descriptions that enable us, in some sense, to compare two different inputs and determine whether they are alike or not, and if not, in what sense they are different, and so on. Detailed descriptions of this kind require structure analysis and syntactic considerations. At this stage we are concerned about shape descriptions that are in some sense primitive and can be made part of more detailed structure analysis.

(a) One simple procedure is to count the number of connected components. Also, determine the number of holes, if any, in the connected components.

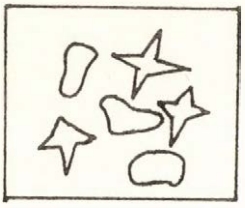
If C is the number of connected components and H the number of holes, then the Euler number E is given by

$$E = C - H.$$

(b) A somewhat more involved description is based on computing metric properties.

(i) Area and Length of Perimeter can be readily computed for digital pictures by simple counting.

Labels can be generated by using these measures and used to sort out subpictures.



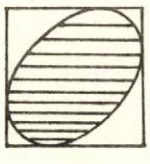
Example Consider a picture made up of blobs and star-shaped objects as shown. It may be possible to separate these on the basis of 'having more interior points' versus 'having more boundary points' as a ratio of the area. (See below, Thinness Ratio).

(ii) The Thinnes Ratio T of a figure with area A and perimeter r is defined as

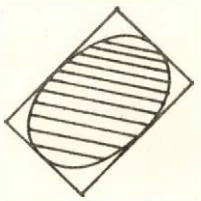
$$T = 4\pi(A/r^2).$$

T has a maximum value which is achieved for a circle. T has been used to characterize figures. Line-like figures have a T-value close to zero, while disc-shaped figures have T-values close to 1, or at least away from zero.

(iii) Aspect Ratio is a measure of the elongatedness of a figure. The aspect ratio of a rectangle is (length/width). For arbitrary shaped figures, use the aspect ratio of their covering rectangles. But to get meaningful values, the orientation of the covering rectangle should be properly chosen. See examples shown below.



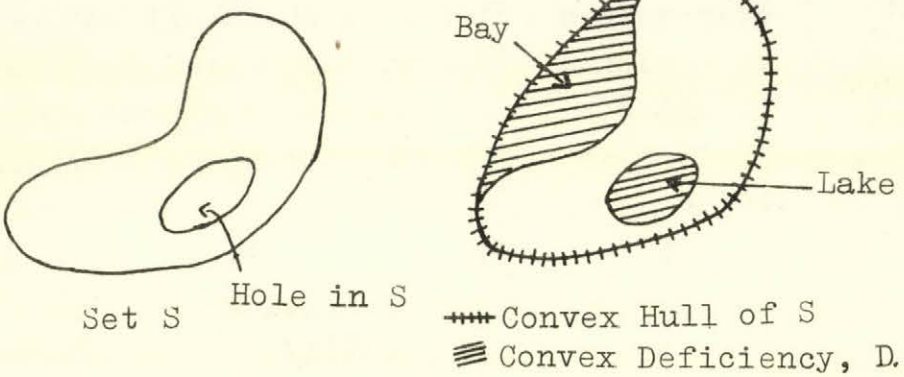
(a) Poor Choice of covering rectangle



(b) Better choice of covering rectangle.

(iv) Descriptions of figures could be based on the "irregularities" that are present.

The Convex Hull H of a set S is the smallest convex set containing S. If S is convex then H=S. The difference H-S is called convex Deficiency D of the set S.



A set is completely defined by its Convex Hull H, and Convex Deficiency D. The components of D come in two varieties: (1) those components lying on the boundary of H: (these are called Bays): (2) Those completely encircled by S: (These are called Lakes).

Convexity of digitized figures must be carefully defined. A circle, for instance, when digitized, consists of a stair-case boundary and may not be considered as convex according to the usual definitions.

Exercise: Given S, write PAX programs to generate H and D of S.

[Note: R.O.Duda & P.E.Hart: Pattern Classification and Scene Analysis; Wiley-Interscience, 1973 has a very readable chapter on "Shape Description". Much of the material above is based on this chapter.]

5. Global Relational Structure Analysis: Generating Picture Descriptions

5.1 So far we have been considering the effectiveness of PAX in preprocessing pictures and subsequently in the feature abstraction stage. We have reviewed several often-used preprocessing procedures like smoothing and sharpening, and several other noise-cleaning and picture enhancement techniques. We have also discussed in some detail the classes of features appropriate to line drawings and to descriptions of single objects.

The question that now arises is this: Having abstracted a set of features how does one proceed further? To answer this question we must recall what we emphasized earlier as being our basic objective in processing pictures. We assume explicitly that input pictures relate to well-defined extra-pictorial situations. We want to analyze the pictures in order to gather information about these situations. The picture descriptions that we want to generate, then, are the kind of descriptions that furnish us information about the situations that the picture is assumed to represent. Clearly, the description must be a structured one. It must say: this picture consists of these and these features, these subsets of features are related in these ways and, hence, presumably represent an object (a situational aspect) of this sort. Those subsets of features similarly represent an object of this other sort. These objects are configured in the picture in these ways and hence represent the following situational aspect. And so on.

It is immediately obvious from the above discussion that relations play a fundamental role in structure analysis at a global level. But they play an essential role even at the level of feature abstraction. The notions: corner, edge, end point, isolated point, disc-like, and so forth are relational notions. But there is a significant difference. Feature abstraction is based on relational considerations at the strictly local level. But at the higher level, relations have to be computed on a more global basis. Also, these relations, valid between complex objects, are, in general, computationally more complex. However, we shall see that a good many of the spatial relations can be computed very easily within PAX.

5.2 Very little systematic work has been done by computer scientists to determine an efficient set of relationships that are applicable to a variety of pictorial situations. The following is a list of relationships that would seem to be essential for the analysis of many classes of pictures:

right of	inside	near
left of	outside	far
above	at the centre of	next to
below	surrounded by	attached to

overlapping (occluding)
isolated (one)
grouped (many)

larger than	longer than	more than
smaller than	shorter than	less than

It should be quite obvious that all these relations can be evaluated efficiently within PAX.

Exercise: Suggest PAX programs for evaluating the above relationships.

A second problem of great and basic importance in relational structure analysis relates to the devising of appropriate data structures to represent these relations. List, Plex and graph structures naturally suggest themselves and have in fact been extensively used, especially, in the context of interactive graphics. Another possibility of course is to use natural language statements, or statements in some appropriately constructed formal language. All these alternatives may have specific advantages in special circumstances. As invariably happens, it may be futile to look for a universal solution. However, whichever alternative is used, it is essential to be able to link up the representation at this level with the representation at the PAX structure level. Easy and natural communication between these two levels of representation is, clearly, a prerequisite to hetararchic processing: (see Sec. 3.2 earlier). This remains very much an unexplored problem.

5.3 One approach to relational structure analysis that has been extensively studied during the last decade is through the use of generative descriptive formalisms for classes of pictures, more commonly known as Picture Grammars, or Picture Languages. A picture grammar relates to a class of pictures and gives explicit

computational rules for generating specific pictures belonging to this class. These rules are generally formalized to resemble grammars of string languager. In the analysis stage, a picture grammar can evidently be used in much the same way as a string-language grammar is used to parse input sentences of that language.

A vast literature exists on the formal aspects of picture grammars, and on parsing procedures that use these grammars. These procedures work reasonably well when one deals with synthetic pictures, i.e., pictures which, in fact, have been generated to begin with using such grammars (explicitly or implicitly). Picture grammars, quite clearly, can play a powerful role in the design of effective graphics languages for interactive graphics. Unfortunately, their potentialities in this direction have not been investigated as well as their intrinsic merit would seem to warrant.

But the use of picture grammars for the analysis of real-life pictures is somewhat more restricted. The reason for this is that real-life pictures form an open-ended class as contrasted with synthetic pictures which form a closed class. In fact, the corresponding picture grammar specifies the necessary and sufficient condition for membership in this closed class. Hence, in parsing such pictures, one need not have to go outside the specifications contained in the picture grammar, in principle. For an open class, on the other hand, such a set of necessary and sufficient conditions does not exist by definition. A syntactic model of an

open class of pictures can only function as a working hypothesis. An analyzer can use such a model as an aid, but not be completely determined by it. Referring back to our earlier discussions about hierarchic and hetararchic processing strategies, an open-ended class requires, in general, hetararchic processing. And the decisions at the various stages have to make use of extrapictorial information (semantics, pragmatics, context, etc) in quite significant ways. This distinction between synthetic and real-life pictures is quite analogous, in detail, to the distinction between formal language texts (such as programs written in a computational language) and natural language texts.

Even in the case of synthetic pictures, it is worth noting that, for purposes of recognition, complete parsing based on a generative grammar need not always be the most efficient. A generative description, although it precisely articulates a procedure for generating a picture, does not explicitly provide information about all the computable properties or relationships embedded in the end result of the generative process. For instance, from the generative description of the letter 'A', it may not be immediately evident that 'A' has a 'corner' of a particular kind at its 'apex'. Recognition of an 'A', on the other hand, may be more readily performed precisely on the basis of such embedded properties and relationships. It may turn out that some of these 'features' are more readily computed by operating directly on the end result (namely, the picture of 'A') rather than trying to infer them from the generative specification (assuming that

such an inference is possible). In this sense, efficient recognition procedures need not be strictly 'inverses' of generative procedures.

In spite of these limitations, as we have seen in detail earlier, complex pictures cannot be recognized except through detailed structure analysis. In so far as generative descriptions provide a systematic means of performing structure analysis, their usefulness in picture processing should not be undervalued. Our earlier remarks merely emphasize the need for constructing generative descriptions that have pragmatic relevance, rather than ones that are merely formally correct. That is, the generative model should enable us to parse the input picture into subpictures, having attributes and relationships that can be directly correlated with the extrapictorial situational details that we are trying to infer by analyzing the picture.

5.4 We shall complete our discussion of picture processing by considering a general meta-level characterization of generative picture languages.

Consider first what aspects a picture language must consist of. (1) There must be a method of describing picture elements (or picture atoms). (2) There must be a method of articulating relationships between picture elements and other constructed fragments. (3) There must be a system of transformations defined over the picture atoms and fragments in terms of which global picture transformations could be realized.

Given a picture language with these aspects, clearly, a picture can be analyzed into its elements and their mutual relationships. Two pictures can also be compared in terms of the transformations that relate their parts.

Keeping these points in mind we shall structure a generative specification language, G, for a class of pictures more precisely now. It is important to note that a particular specification language necessarily has to be tied to some specific action mechanism. The action primitives incorporated in G would explicitly be related to the action repertoire of this mechanism that is to be used in realizing the specific pictures belonging to the class delimited by G.

A Specification Language G for a class of pictures is a 5-tuple: G=G(P,A,R,C,T), where P is a set of primitives; A is a set of attributes; R, a set of relations; C, a set of composition rules; and T, a set of transformations.

P consists of a set of primitive actions the performance of which generate primitive picture atoms. These atoms can be identified by the same names as the primitives that generate them.

Each primitive (and, hence, the atom that it generates) has associated with it a set of attributes, i.e., specificational parameters, which can assume values from well-defined ranges. Fixing the values of all (or some) of the attributes of a primitive results in an assignment (or partial assignment) to the primitive. A primitive with an assignment (partial assignment)

generates a picture atom with all (some) of its attribute values determined.

P and A together, thus, allow us to generate picture atoms with particular assigned properties. We now require a machinery in G that would allow us to put these atoms together to form larger picture fragments. The composition rules precisely allow us to accomplish this. These rules are specified making use of the set of relations R.

Each relation is an m-ary predicate defined over the attribute values of the constituents of picture fragments.

Let p_1, p_2 be primitives with assigned properties. Let r be a relation defined over some subset of properties of p_1 and p_2. Then $r(p_1,p_2)$ is a picture fragment whose constituents are the atoms p_1 and p_2 such that their assigned properties satisfy the relation r. Let us call this fragment f. Then a composition rule has the form:

$$f \longleftarrow r\,(p_1,p_2).$$

To complete the specification of this composition rule we must identify the attributes of f. We shall indicate this by writing the c-rule as follows:

$$f(\alpha_1,\ldots,\alpha_i) \longleftarrow r\,(p_1(\ ),p_2(\ )),$$

where the $\{\alpha_j\}$ are the attributes of f. Notice that the $\{\alpha_j\}$ need not be the same as the attribute sets applicable to p_1 and p_2. In general, they will be functions computable making use of the attribute values on the right-hand side as arguments.

Example: Let p_i be Vertical & Horizontal lines. Attribute sets of p_i are length, thickness coordinates of end-points, etc. f could be a rectangle composed out of the p_i. f could then have attributes, area, perimeter, centre of gravity, etc, which are not applicable to the p_i. But all these attributes of f are obviously computable from those of the p_i.

To proceed with higher-level compositions, we should permit the occurrence of fragments on the right-hand side of c-rules. So we can generalize a c-rule to have the form:

$$f_3(\ ) \longleftarrow r\left(f_1(\ ), f_2(\ )\right)$$

where, in particular cases, f_1, f_2, or both could be primitives.

The last component of G is the set of transformations, T. We can distinguish two kinds of transformations:

(1) Transformations that alter properties of a given picture fragment without spatial change. Coloring, texturing, shading, are of this type. These operations give information about the material properties of the pictorial objects, and the lighting conditions.

(2) Transformations that delete components of a fragment or introduce spatial changes in parts or the whole of a fragment. Deletion, translation, rotation, projection, mirror inversion, contraction, dilatation, distortion, are transformations of this type.

Transformations, in general, are computational procedures that take picture fragments into picture fragments, and use the attribute values of input fragments as arguments for computing the output fragments. Such procedures could be quite complex in structure.

5.5 In a formal definition of G, notational conventions would have to be introduced so that primitives with attribute assignments, relationships, and transformations can be explicitly represented, and can be uniformly interpreted into action sequences that result in the creation and manipulation of picture fragments. Such a notational scheme would also include a formal definition of a program, or a procedure, in the specificational language, G. To enable this, as we noted in the case of PAX earlier, sequencing conventions and conditionals would have to be included in G.

The way we have defined a specificational picture language, G, makes it an exact analogue of a computational (i.e., programming) language. The general power of a programming language is needed to generate arbitrarily complex pictures. Simpler picture grammars that have been extensively studied making use of context Free and other types of composition rules will be seen to be very special cases of the general metaformalism that we have outlined. The theory of picture grammars has exactly the same relationship to general picture languages, as automata and formal language theories have with respect to general programming languages. In both cases the grammars allow us to order the class of languages in revealing ways according to complexity

measures that can be explicitly stated.

5.6 This completes our brief discussion of picture processing using PAX.

A Selected list of references in
picture processing by computers
and some related topics

(Note: Decision Theory not included)

Books & Conference Proceedings

1. H.C.Andrews: Computer Techniques in Image Processing; Academic Press, 1970.

2. G.C.Cheng et al (Eds): Pictorial Pattern Recognition; Thompson Book Co, Washington D.C., 1968.

3. R.O.Duda & PE Hart: Pattern Classification and Scene Analysis; Wiley-Interscience, 1973.

4. A.Grasselli (Ed): Automatic Interpretation and Classification of Images; Academic Press, 1969.

5. L.N.Kanal (Ed): Pattern Recognition; Thompson Book Co., Washington D.C., 1968.

6. S.Kaneff (Ed): Picture Language Machines; Academic Press, 1970.

7. P.A.Kolers & M.Eden (Eds): Recognizing Patterns; MIT Press, 1968.

8. B.S.Lipkin & A.Rosenfeld (Eds): Picture Processing and Psychopictorics; Academic Press, 1970.

9. M.Minsky & S.Papert: Perceptrons, An Introduction to Computational Geometry; MIT Press, 1969.

10. F.Nake & A.Rosenfeld (Eds): Graphic Languages; North-Holland, 1972.

11. A.Rosenfeld: Picture Processing by Computer; Academic Press, 1969.

12. W.Wathen-Dunn (Ed): A Symposium on Models for Perception of Speed & Visual Forms MIT Press,1967.

On Computer Graphics

13. W.M.Newman & RF Sproull: Principles of Interactive Computer Graphics; McGraw-Hill, 1973.

On Visual Perception

14. P.Brodatz: Textures; Dover Publications, 1966.

15. JJ Gibson: The Perception of the Visual World; Houghton-Mifflin, Boston, 1950.

16. RL Gregory: Eye & Brain; World Univ. Library, McGraw-Hill, 1966.

17. RL Gregory: The Intelligent Eye; McGraw-Hill, 1970.

18. B.Julez: Foundations of Cyclopean Perception; Univ. Chicago Press, 1971.

B. Review Articles & Collections in Journals

1. HC Andrews & LH Enloe (Eds): Special Issue on Digital Picture Processing; Proc. IEEE, 60, July, 1972.

2. HB Barlow, R.Narasimhan & A Rosenfeld: Visual Pattern Analysis in Machines & Animals; SCIENCE, 177, Aug. 1972, 567-575.

3. K.S.Fu & PH Swain: On Syntactic Pattern Recognition; in JT Tou (Ed): Software Engineering, Vol. 2, pp. 155-182; Academic Press, 1971.

4. K.S.Fu (Ed): Special Issue on Syntactic Pattern Recognition. Part I and Part II; Pattern Recognition, 3, Nov. 1971, and 4, Jan. 1972.

5. L.D.Harmon (Ed): Special Issue on Digital Pattern Recognition; Proc. IEEE, 60, October, 1972.

6. MD Levine: Feature Extraction: A Survey; Proc. IEEE, 57, Aug. 1969, 1391-1407.

7. WF Miller & AC Shaw: Linguistic Methods in Picture Processing: A Survey; Proc. FJCC, Dec. 1968, 279-290.

8. A.Rosenfeld: Picture Processing 1972; Tech.Rept. TR-217, Computer Science Center, Univ. Maryland, Jan. 1973.

9. A.Rosenfeld: Progress in Picture Processing 1969-1972; Computing Surveys, 5, March, 1973. (ACM Publication).

C. Papers

1. DV Ahuja & SA Coons: Geometry for Construction and Display; IBM Systems Jour. No. 3 & 4, 1968, 188-205.

2. W.J.Bouknight: A Procedure for Generation of 3-D half-tone computer graphics presentations; comm. ACM, 13, Sep. 1970, 527-536.

3. CR Brice & CL Fennema: Scene Analysis Using Regions; Artificial Intelligence, 1, 1970, 205-226.

4. MB Clowes: On Seeing Things; Artificial Intelligence, 2, 1971, 79-116.

5. A.S.Coons: Surfaces for the Computer-aided design of forms; Project MAC Rept., MAC-TR-41, June, 1967.

6. ES Deutsch & NJ Belknap: Texture Descriptors Using Neighbourhood Information; Computer Graphics & Image Processing, 1, Aug. 1972, 145-168.

7. G.P.Dinneen: Programming Pattern Recognition; Proc.WJCC, March 1955, 94-100.

8. RD Duda & PE Hart: Use of Hough Transformation to detect Lines & Curves in Pictures; Comm.ACM 15, Jan. 1972, 11-15.

9. G.Falk: Interpretation of Imperfect Line Data as a 3-D Scene; Artificial Intelligence, 3, 1972, 101-144.

10. J.Feder: Languages of Encoded Line Patterns; Information & Control, 14, 1969, 9-52.

11. H.Freeman: On the Encoding of Arbitrary Geometric Configurations ; IEEE Trans. EC-10, June, 1961, 260-268.

12. H.Freeman & JM Glass: On the Quantization of Line-drawing Data; IEEE Trans. Sys.Sci.Cyb., SSC-5, Jan. 1969, 70-78.

13. S.B.Gray: Local Properties of Binary Images in 2-D; IEEE Trans. C-20, May 1971, 551-561.

14. U.Grenander: Foundations of Pattern Analysis; Quart. Applied Matts., 27, Apr. 1969, 2-55.

15. A.Guzman: Decomposition of a Visual Scene into 3-D Bodies; Proc. FJCC, 33, 1968 , 291-304.

16. GT Herman: Two Direct Methods for Reconstructing Pictures from their Projections: a Comparative Study; Proc. SJCC, May. 1972, 971-984.

17. F.Holdermann & H.Kazmierczak: Preprocessing of gray-scale pictures; Computer Graphics & Image Processing, 1, Apr. 1972, 66-80.

18. RA Kirsch: Computer Interpretation of English Text and Picture Patterns; IEEE Trans. EC-13, Aug. 1964, 363-376.

19. K.Knowlton & L.Harmon: Computer-produced Gray Scales; Computer Graphics & Image Processing, 1, Apr. 1972, 1-20.

20. PP Loutrel: A Solution to the Hidden-line Problem for Computer-drawn Polyhedra; IEEE Trans. C-19, March, 1970, 205-213.

21. R.Mahl: Visible Surface Algorithms for Quadric Patches; IEEE Trans. C-21, Jan. 1972, 1-4.

22. PC Maxwell: The Perception & Description of Line Drawings by Computer; Computer Graphics & Image Processing, 1, Apr. 1972, 31-46.

23. R.Narasimhan: Labelling Schemata & Syntactic Descriptions of Pictures; Information & Control, 7, June, 1964. 151-179.

24. R.Narasimhan & JP Fornango: Some Further Experiments in the Parallel Processing of Pictures; IEEE Trans. EC-13, Dec. 1964, 748-750.

25. R.Narasimhan: Syntax-directed Interpretation of Classes of Pictures; Comm. ACM, 9, March, 1966, 166-173.

26. ME Newell et al: A Solution to the Hidden Surface Problem; Proc. ACM National Conf., Aug. 1972, 422-431.

27. FI Parke: Computer-generated Animation of Faces; Proc. ACM. National Conf., Aug. 1972, 451-457.

28. JL Pfaltz: Web Grammars & Picture Description; Computer Graphics & Image Processing, 1, Aug. 1972, 193-220.

29. TC Rind fleisch et al: Digital Processing of the Mariner 6 and 7 Pictures; J.Geophysical Research, 76, Jan. 1971, 394-417.

30. L.G.Roberts: Machine Perception of 3-D Solids; Tech. Rept. 315, Lincoln Lab., MIT; May, 1963.

31. A.Rosenfeld J.L Pfaltz: Distance-functions on Digital Pictures; Pattern Recognition, 1, July 1968., 33-62.

32. A.Rosenfeld: Connectivity in Digital Pictures; Jour.ACM, 17, Jan. 1970, 146-160.

33. A.Rosenfeld & M.Thurston: Edge & Curve Defection for Visual Scene Analysis; IEEE Trans. C-20, May. 1971, 562-569.

34. T.Sakai et al: Line Extraction & Pattern Detection in a Photograph; Pattern Recognition, 1, March 1969, 233-268.

35. A.C.Shaw: A Formal Picture Description Scheme as a Basis for Picture Processing Systems; Information & Control, 14, 1969, 9-52.

36. W.Stallings: Recognition of Printed Chinese Characters by Automatic Pattern Analysis; Computer Graphics Image Processing, 1, Apr. 1972, 47-65.

37. J.Warnock: A Hidden-line Algorithm for Half-tone Picture Representations; Tech. Rept. 4-5, Dept. Comp. Science, Univ. Utah, May. 1968.

38. R.A.Weiss: Be Vision, A Package of IBM 7090 FORTRAN Programs to draw Orthographic Views of Combinations of Plane & Quadric Surfaces; Jour.ACM, 13, Apr.1966, 194-204.

39. C.Wylie et al: Half-tone Perspective Drawings by Computer; Proc. FJCC, 31, 1967, 49-58.

Addenda

40. U.Montanari: A Method of Obtaining Skeletons using a Quasi-Euclidean Distance; JACM, 15, (1968), 600-624.

41. U.Montanari: Continuous skeletons from Digitized Images; JACM, 16, (1969), 534-549.

42. U.Montanari: A Note on Minimal Lengths Polygonal Approximation to a Digitized Contour; Comm. ACM, 13, (1970), 41-47.

43. U.Montanari: On Limit Properties in Digitized Schemes; JACM, 17 (1970), 348-360.

44. H.Blum: A Transformation for Extracting New Descriptors of Shapes; in A(13), pp. 362-380.

A TANULMÁNYOK sorozatban eddig megjelentek:

1/1973 Pásztor Katalin: Módszerek Boole-függvények minimális vagy nem redundáns, $\{\wedge, \vee, \neg\}$ vagy {NOR} vagy {NAND} bázisbeli, zárójeles vagy zárójel nélküli formuláinak előállitására

2/1973 Вашкеви Иштван: Расчленение многосвязных промышленных процессов с помощью вычислительной машины

3/1973 Ádám György: A számitógépipar helyzete 1972 második felében

4/1973 Bányász Csilla: Identification in the presence of drift

5/1973* Gyürki J.-Laufer J.-Girnt M.-Somló J.: Optimalizáló adaptiv szerszámgépirányitási rendszerek

6/1973 Szelke Erzsébet-Tóth Károly: Felhasználói Kézikönyv /USER MANUAL/ a Folytonos Rendszerek Szimulációjára készült ANDISIM programnyelvhez

7/1973 Legendi Tamás: A CHANGE nyelv/multiprocesszor

8/1973 Klafszky Emil: Geometriai programozás és néhány alkalmazása

---

*-gal jelölt kivételével a TANULMÁNYOK megrendelhetők az Intézet Könyvtáránál /Budapest, I. Uri u. 49./

1909