Principles of the Parallel Shape Coding with the Termination Condition

Zbigniew M. WÓJCIK, Barbara E. WÓJCIK
The Division of Mathematics, Computer Science and Statistics,
The University of Texas at San Antonio, San Antonio, TX 78285

ABSTRACT. *Elementary features are detected by calculating the number of objects inside partly overlapping windows fixed in an image plane. Each window's contents is processed by a separate processing element (PE) on a SIMD grid or pyramid architecture. Two neighboring feature chunks are merged by adjacent PEs, and the merged feature chunks are joined in each next lth parallel step by every 2^lth PE possessing complementary and adjacent feature. The shape coding terminates if a complete set of features (such as curve bounds, tips, corners, segments, regions forks and junctions) making a whole object meet together on one PE. This complete set is defined and proved to be sufficient to terminate the shape coding. Hierarchy of elementary features for a merging process is established to recognize object bounds and more primitive features first to have a complete feature set adequate for consideration by the termination condition at a given level of parallel image recognition (i.e.,* primal sketch, $2\frac{1}{2}D$ sketch *and* 3D world model). *The approach has the property of mapping of image segments directly into phrases and English sentences.*

1. Introduction

Many present day computer vision systems analyze shapes sequentially. Can a computer see the entire picture at once? If so, how one processor analyzing one small image fragment will know about the processing performed by many other processors attached to all other image fragments? The first pyramid computers process in parallel [8, 16, 19] pieces of contours detected by template matching. We approach parallel shape analysis using a SIMD grid architecture by merging elementary features detected during the calculation of the number of objects within a window.

Propagation component labeling [2] determines, in parallel, the connectedness of pixels of the same gray level (assuming that each pixel has its own processor). Given an $N \times N$ image, the row-major label $Ni + j$ with $i, j = 0, 1, ..., N-1$ is assigned to each image pixel. Two neighboring pixels (or subregions) of the same (or similar) gray level receive the same (the smallest) label. Initially, only the pixel at $(0, 0)$ has its own label if it is of a grey level looked for. All adjacent pixels with the same gray level will assume that label in each new iteration. The final label is the smallest label of any of the connected members. Execution is complete when there are no more label changes. Before propagation labeling is completed, it is not known which label is the minimum (i.e., permanent). Labeling of a spiral (the worst case) will take N^2 steps.

Propagation component labeling does not provide any information about shape. It is also sensitive to accidental

discontinuities. The re-assignment of the labels in each new step is time consuming. One improvement would be to record addresses of all pixels tentatively labeled (i.e., in the previous step), and then to propagate the new minimum label to the known addresses of PEs. However, very long strings of addresses would have to be broadcast to PEs together with the labels.

It will be shown in this paper, that shape codes can be handled more efficiently than by recording and broadcasting the addresses of pixels. Shape can be recognized directly from window gray levels. Our algorithm with shape coding is faster than the propagation component labeling: it does not waste time re-labeling pixels and broadcasting addresses of individual pixels. Instead, shape codes are transmitted carrying all the necessary information about the shape components. Information about individual pixels is involved implicit in shape code.

Sollin's component algorithm for graphs [6] uses a *divide-and-conquer* strategy. This algorithm is based on a quadtree decomposition on image plane. The image is divided into quadrants. Each quadrant (greater than 1 pixel) is labeled recursively (i.e., by dividing each subsequent quadrant into quadrants up to 1 pixel quadrant). Minimum label is passed through the adjacent quadrant boundaries in each recursive step. The *divide-and-conquer* strategy is of a logarithmic complexity, but each re-labeling step is very long, as in the propagation component labeling. Re-labeling takes place at fixed quadrant boundaries. Some PEs can specialize in broadcasting labels: single pixel quadrants communicate with single pixels, 4-pixel quadrants with 4-pixel quadrants, 4^n-pixel quadrants with 4^n-pixel quadrants. Broadcast of addresses of pixels having uniform labels can be efficiently organized, e.g., incorporating a pyramid architecture. But addresses of all individual pixels carrying the same label must still be handled.

Methods for finding extreme pixels in each quadrant may ignore a discontinuity inside a quadrant and as a result may merge two or more disjoint objects into one entity, a drawback of the *all-components extreme point approach* [20, 21]. Improving this approach, extreme points of separate objects in a quadrant should be found as a result of segmentation by the divide-and-conquer strategy or by propagation component labeling. Extreme points of individual objects can be found sequentially by an effective algorithm in a contour tracking process [25].

Neither the divide-and-conquer strategy nor the propagation component labeling are capable of coding shapes. Even the improved version (suggested above, by propagation of addresses for re-labeling) can handle only addresses of individual pixels, because shapes of re-labeled blobs are not coded. The approaches count adjacent pixels of a similar gray level only and extract them from an image: in addition, they must be followed by a shape approximation process if a shape representation is required.

A pyramid technique for parallel detection of smooth curves [8] depends on a bottom-up analysis of the adjacency of elementary curve elements in neighboring blocks (windows) by a higher-level pyramid cell. Each cell of the pyramid records the position and slope of a curved strip and positions of endpoints of the strip. Higher-level cells record more distant endpoints (since a larger block of input image is covered by a higher-level cell). Some higher-level cell will be a root recording a consistent smooth curve. The pyramid architecture designates certain cells in advance to identify many possible occurrences of features. The coordinates of endpoints go directly up the pyramid making a computational burden for some higher level pyramid cell if several curves are detected in the same large field belonging to the cell. The time complexity for such computations is $O(number\_of\_curves)$. For instance, two nearly parallel curves must be handled by the same cell at some level. This higher-level cell will be a bottleneck for more lines found in its field. A search is needed to match endpoints of strips found in adjacent subfields at a lower level. Doubling back of the same curve will also require the same cell at some level for identification.

In a process of collecting features such as forks, junctions, regions, corners and curve bounds connected through edges, a node is capable of making a decision whether to

look around for some missing features in the image or to complete the image recognition. Classes of objects are composed of some specific numbers of these features. For example, polygons have only one region. A corner view of a cube has three regions, four forks, nine edges and three corners. The termination condition for a recognition process may be based on the graph theory concepts using some relations between object components. *Euler's Theorem* [5, 15] defines relation between the number e of edges, the number v of vertices in a planar graph, and the number r of regions (including one unbounded region external to the graph), using the equation: $r = e - v + 2$. The *Handshaking Theorem* $2e = \sum_{v \in V} deg(v)$ [5, 15] expresses the relation between the number of edges e and the degree deg of all vertices V making a graph. The above two Theorems can be used in computer vision as a termination condition after disregarding an unbounded region and assigning a meaning 'corner' to a vertex of degree 2, 'fork' to a vertex of degree 3, etc., and 'object edge' to graph edge, and 'image region' to the graph region.

Euler's Theorem does not classify vertices (i.e., does not distinguish between corner, tip, fork, junction, etc.), and the *Handshaking Theorem* does not consider regions, which would be disadvantageous for a termination condition of recognition procedures in computer vision. Combination of them both treats all vertices as equally significant. However, in computer vision the line tips are the least meaningful and should be easily disregarded if a contour is blurred in the image. In general, recognition levels called *primal sketch*, $2\frac{1}{2}D$ *sketch* and *3D world model* are distinguished in literature [3, 14, 18, 22, 23] and should be considered by termination conditions in shape processing.

We describe relations between the numbers of lines (i.e., straight segments or curves) and corners, forks, junctions and regions for computer vision purposes in a way different from that used in the literature [3, 10, 14, 18, 22, 23] . Tips are not considered explicit in our termination conditions, but any line is constrained to being delimited by either tips, or corners, or curve bounds, or forks, or junctions or by combinations on them. In this way tips are implicit in the number of lines in our termination conditions.

As in [8], the merging of adjacent features is performed by the nodes one level higher. The image plane is divided into quadrants. We propose a SIMD architecture similar to [8], with the following differences: a) We use special termination conditions for every node to know whether the parallel shape coding is completed when having some set of features. For instance, the resulting approximation is sent one level up if a complete set of features has not been found yet. b) A few termination conditions are set up, each appropriate for making the *primal sketch*, $2\frac{1}{2}D$ *sketch* and 3D *world model* in parallel. c) The termination condition for the *primal sketch* has the highest priority. The $2\frac{1}{2}D$ *sketch* has a lower priority, and the lowest priority is for the 3D (3-dimensional) *world model*. d) The detection of the elementary features is done by calculation of the window objects instead of by template matching. e) The approach has the property of mapping image segments directly into words and phrases.

2. Feature Detection and Feature Merging

Elementary features are detected in a window using the numbers of window objects [24-26]. Different combinations of the numbers are characteristic to specific image features (as segment, fork, tip, junction, discontinuity). For instance, the name "segment" is achieved if the number I_w of objects equals 1 inside the window and the number I_o of background parts is 2 (Fig. 1). For a junction, I_w=1 and I_o=4. For a fork, I_w=1 and I_o=3. The same numbers are obtained for all possible rotations of the same feature inside the window and for variable thicknesses of the feature. Only one single analysis of window contents (gray levels) is necessary.

In our method, elementary features, especially segments, detected in parallel by PEs of a grid or a pyramid net are merged into chunks by higher level nodes. When merging, every two consecutive nodes are replaced by one node when the corresponding segment slopes are

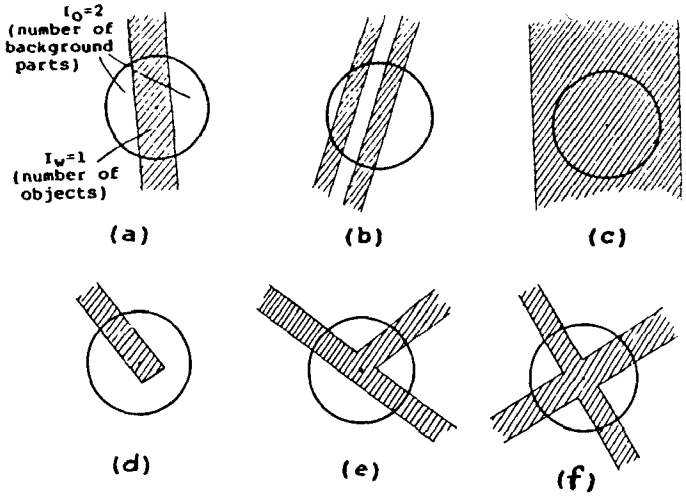


Fig.1. Window approximation by mapping of window gray levels into feature name using the number I_w of window objects and the number I_o of window background parts. The symbolic window approximations are: a) segment, b) two segments; c) interior; d) tip; e) fork; f) junction

similar. Reduction of elementary segments to a straight line or to a curve is performed in this way (Fig.2). The merged chunks are united in turn into greater pieces, and so on, until the entire shape has been coded in parallel [9]. Maximum of $2^{n/l}$ merging operations can be done in each subsequent lth parallel step for a linear feature occupying $2^n = N$ contiguous PEs. In order to know where to look for some missing chunks to match some already detected object fragments to one of the termination conditions, slopes of the possessed feature pieces are used. The missing feature chunks are looked for in the expected directions. Knowledge on the maximum size of the pieces looked for is also available at a given stage of parallel processing: feature size does not exceed the quadrant size belonging to the node at a current level of processing.

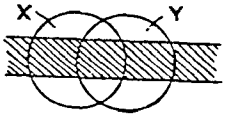
In our method, names of features and their parameters (including the count of pixels of the same property) are handled instead of propagating a label having nothing in common with the shape name. In particular, a rough set composed of elements consisting of feature names with attached attributes and parameters [24] is broadcast to fixed-address nodes (PEs). The receivers identify more complex features, based on feature adjacency, expectations and constraints. A rough approximation of larger pieces of shapes is done to reduce to a minimum the amount of information representing an object. The rough approximations deal both with symbols (names of features and objects) and with quantitative representations (e.g., with positions, slopes and dimensions of features). Instead of a long process of re-labeling of many pixels, merging of features is performed. A *rough grammar* [29, 30] can be applied to support the divide-and-conquer strategy. A data-driven execution can guide the parallel shape coding: every leaf PE will wait until the window gray levels will become available for the detection of elementary features.

$$rA(X+Y) = \underline{A}X = \underline{A}Y = \underline{A}(X+Y) = \text{segment, if } s_X \approx s_Y \text{ and } \underline{A}X = \underline{A}Y = \text{segment.}$$

where:

$$s_X \approx s_Y \iff |s_X - s_Y| \leq \Delta s ;$$

r denotes the reduction; Δs is an admissible tolerance for s (slope).



$$rA(X+Y) = \underline{A}X + \text{attributes} * \text{parameters} =$$

$$= \underline{A}X + (\text{mass,position,length,slope}) * (m,(x,y),l,s),$$

$$\text{if } s_X \approx s_Y \text{ and } \underline{A}X = \underline{A}Y = \text{segment.}$$

where:

$$m = m_X + m_Y$$
$$x = (x_X + x_Y)/2$$
$$y = (y_X + y_Y)/2$$
$$l = l_X + l_Y$$
$$s = (s_X + s_Y)/2$$

No reduction:

$$rA(X+Y) = AX + AY, \quad \text{if } \underline{A}X \neq \text{segment or } \underline{A}Y \neq \text{segment.}$$

Fig.2. Symbolic approximation $\underline{A}(X \cup Y)$ of two partly overlapping windows containing segment and approximation $A(X \cup Y)$ of the reduced two-window segment involving both the symbolic and a required quantitative window approximation

3. A Parallel Coding of Edges and Regions

A rooted tree structure is used. At the bottom level of parallel processing, the leaves are the contents of the windows fixed in the image plane, and then the nodes are the ancestors of the lower level sub-trees. Data received from the 4-children is used for merging of segments, tips, curve bounds, forks and junctions, if the object approximation is not complete. A complete shape approximation is transferable to a host.

At the bottom (first) level, all the nodes (PEs) detect their elementary features in parallel by calculating the numbers of window objects [24-26]. Each window (Fig. 3) is under the control of a separate PE. Feature name and its particular quantitative parameters (e.g., segment coordinates, slope, length, mass and thickness) are detected by each PE. The same pixels will belong to a number of overlapping windows. To prevent a single object fragment from multiple processing by several PEs, each window sends its feature approximation to its neighbors in directions different from the slope of the possessed segment. Among the selected neighbors sharing the same object fragment, one window is chosen having the highest I_w (the number of objects). From among a few overlapping windows with the same (highest) I_w, a window is chosen with a feature of the maximum mass.

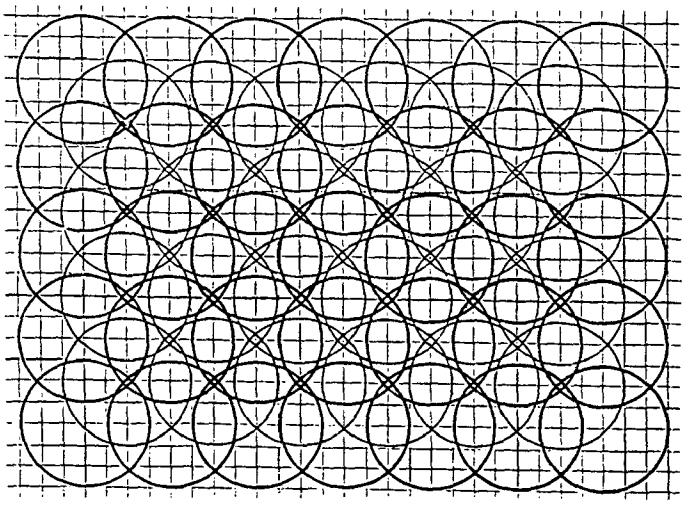


Fig.3. Overlapping windows for parallel shape coding

Any PE staying active sends its feature approximation to its parent. This feature approximation is merged by the receiver to feature chunks received from other children. The result of merging is sent to the ancestor examining its four children, provided coding of the entire object was not completed yet. Getting a full set of the features, a parent completes the object recognition.

Example 1. PEs#(4,11) and #(5,10) send their elementary features to their parent PE#(5,11) at the first level of parallel processing of OBJECT 5 in Fig.4. The PE#(5,11) does not have two tips of the possessed chunk, so it does not announce recognition of a whole object, but sends its approximation to its ancestor, PE#(7,11).

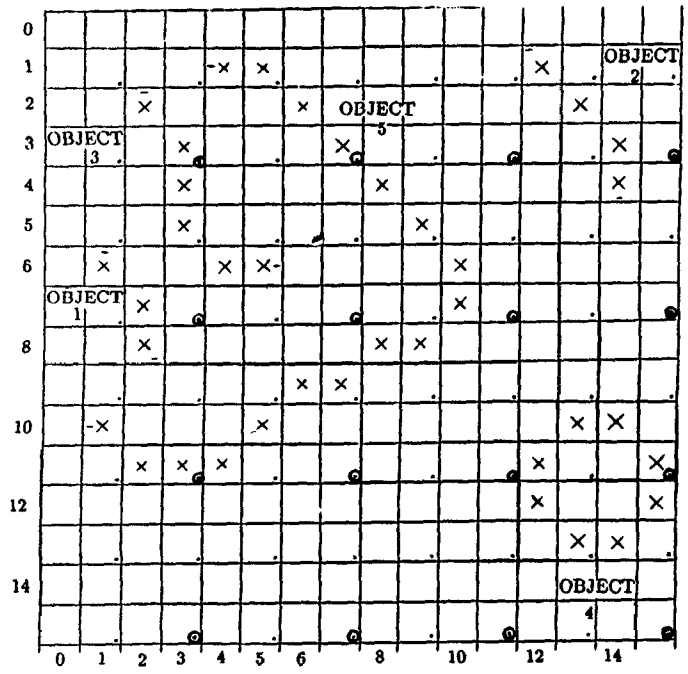


Fig.4. Examples of simple drawings on a grid configuration. Each square represents a PE. To involve the functions of a pyramid, some selected grid nodes are assigned also the functions of nodes of higher level pyramid nodes. Squares with one dot represent both the leaves and the second level nodes of a pyramid, squares with one dot and circle represent both the leaves, the second level nodes and the third level nodes

Example 2. PE#(2,8) in Fig.4 sends its tip approximation to the parent PE#(3,9). At the 3rd level of parallel processing, the PE#(3,7) merges the tip detected by the #(1,6) (and received through 2nd level #1,7)) to the segment detected by PE#(2,7). The merging of the three element curve (OBJECT 1) involves 5 levels of parallel processing on a pyramidal architecture, because OBJECT 1 is located on the border of different quadrants. The 4th level PE#(7,7) still will not receive the tip detected by #(2,8), but the 5th level PE#(15,15) will announce recognition of OBJECT 1, because one segment continuous with two tips constitutes a full set of features for the simplest object.

In general, the results of merging are sent to a nearest doubly odd PE # $(2^l i - 1,\ 2^l j - 1)$, $(2^l i - 1,\ 2^l j - 1) \in \{\{1,2,...,N-1\},\{1,2,...,N-1\}\}$, $N = 2^n$, on a grid configuration, where merging is done on any couple of adjacent segments (Fig.2) by execution of $a(f_{own} \cup f_{neighbor}, s)$, where s is a merging operation, f is a feature approximation [24], a is a requested approximation of merged features (qualitative $\underline{A}$ or quantitative A, see Fig.2), and l is the level of parallel processing. Merging is skipped if two feature chunks do not match each another.

Example 3. At the third parallel processing level (Fig.4), the ancestor PE#(7,7) merges three feature chunks received from its children PEs#(3,3), #(3,7) and #(7,7). The chunks involve two tips and make a full set of features, so PE #(7,7) announces identification of the entire curve (OBJECT 3).

Example 4. On the second level of parallel processing, the nodes #(13,1) #(13,3) and #(15,3) send approximations of their chunks to their parent PE#(15,5). The 3rd level PE#(15,5) and PE#(15,7) send their features to 4th level PE#(15,7) which performs merging and having two tips and a one line announces identification of the curve (OBJECT 2).

A formal specification of the termination condition for parallel shape coding is helpful for appropriate control of execution of programs by nodes (e.g., to know, when segmentation of a single object is completed when having several distinct objects).

THEOREM 1. Parallel coding of shapes involving only tips, straights, curves and corners (without regions, forks or junctions) is completed, if:

$$L = C + 1, \tag{1}$$

where C is the number of corners or curvature bounds (such as upper, lower, left and right bounds), L is the number of curves or segments with two endings (e.g., connecting: corner - corner, bound - bound, corner - tip, bound - tip, corner - bound, tip - tip).

PROOF (inductive). Consider Figs. 4 and 5.a as examples of simple drawings. Having no corners nor curvature bounds we have $L = 1$ (i.e., in the basis step there is only one tip - tip connection). When incrementing L by one, the first and the last segment (or curve) must have a tip (because in this THEOREM a drawing is assumed to be without forks, junctions and regions). In the inductive step let us notice, that any new single corner or curvature bound (denoted by C) increments L by one, and then the true equation $(L+1) = (C+1) + 1$ reducing to Eq.(1) is achieved when assuming the truth of the inductive hypothesis (Eq. (1)). •

a) b)

Fig.5. A few simple drawings satisfying: a) Eq.(1); b) Eqs.(2) and (4) (the last 3D object with shading having: L=24 lines, C=5 corners, R= 7 regions, F=9 forks, $J = |features_4| = 1$ junctions and $|features_5| = 1$ and yielding: $L = 24 = |features_5| * 4 + J * 3 + F * 2 + C * 5 + 1 - 7 = 1 * 4 + 1 * 3 + 9 * 2 + 5 + 1 - 7$

Any element which does not have the entire object (i.e., with a complete set of tips, curvature bounds and

lines (THEOREM 1)) sends the possessed object approximation one level up to its ancestor.

Example 5. OBJECT 5 in Fig.4 has one corner (or right bound represented by PEs #(10,6) and #(10,7)) two tips and two lines, so $L = 1+1 = 2$. The 5th level ancestor (root), PE#(15,15), receives two tips from 4th level children, PEs #(7,7) and #(7,15), and detects one right bound based on approximations received from 4th level PEs #(15,7) and #(15,15). The 4th level PE#(7,15) does not announce identification of an entire object, because it has only one tip. The 5th level PE#(15,15) has a complete set of features, hence it announces recognition of the entire OBJECT 5.

Many horizontal or vertical segments do not form object bounds and then they do not affect the counters C and L in Eq.(1). Any horizontal segment ended by a non-homogeneous ending (e.g., either by a left-lower and right-upper bounds (Fig.6.a), or by upper-left and lower-right bounds (Fig. 6.b)) does not form curve bounds and as such does not increment L nor C.

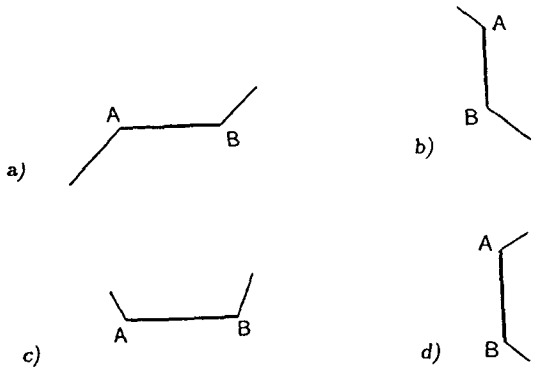


Fig.6. Different type endings A, B of a flat segment: a), b) non-homogeneous signify no curve bound; c), d) homogeneous endings signify a curve bound

LEMMA 1. Any flat segment (horizontal or vertical) delimited by a nonhomogeneous ending (see Fig.6) can be taken apart from a curve bound.

PROOF. Let A and B be nonhomogeneous endings of a flat segment. Approximations of the two nonhomogeneous endings meet at some level of parallel processing contradicting a bound of a horizontal segment (Fig. 6.a) or a bound of a vertical edge (Fig.6.b). Only homogeneous endings (i.e., either lower-left and lower-right, or upper-left and upper-right (Fig.6.c), or upper-right and lower-right (Fig.6.d), or upper-left and lower-left) meeting at some node of the processing tree and delimiting the same segment form a curve bound. •

LEMMA 2. Complexity of the detection of a flat nonhomogeneous segment or a curvature bound is log_2 $(flat\_segment\_length)$ or log_2 $(curve\_bound\_length)$ steps.

PROOF. Endings of the same flat segment (hypothesized bound) occur to be contradictory after log_2 $(flat\_segment\_length)$ steps, and information about this contradiction is utilized or transferred up instantly. Information about noncontradictory segments is processed in the same way. •

Any horizontal or vertical segment may be coded in parallel according to the method presented in [28] (see Fig. 7). The described above parallel processing associated with the *primal sketch* goes through levels until the entire continuous linear object has been collected (i.e., until Eq.(1) is satisfied), accessing the children distant by 2^l or by $\sqrt{2^{2l}+2^{2l}}$ nodes in each lth level of processing. Any fetched feature $chunk1$ is merged with the matching possessed feature $chunk2$ by executing $a(chunk1 \cup chunk2, s)$, where a denotes the symbolic or quantitative approximation of the operands listed in the parentheses (compare Fig.2), and s is the merging operation.

THEOREM 2. Parallel coding of shapes involving tips, straights, curves, corners and regions and without forks or junctions is completed, if:

$$L = C + 1 - R, \tag{2}$$

where R is the number elementary regions, L and C have the same meanings as in Eq.(1).

PROOF (inductive). For an object without regions, forks and junction the Eq.(2) reduces to Eq.(1). For an object consisting of one region only (i.e., for a circle) we

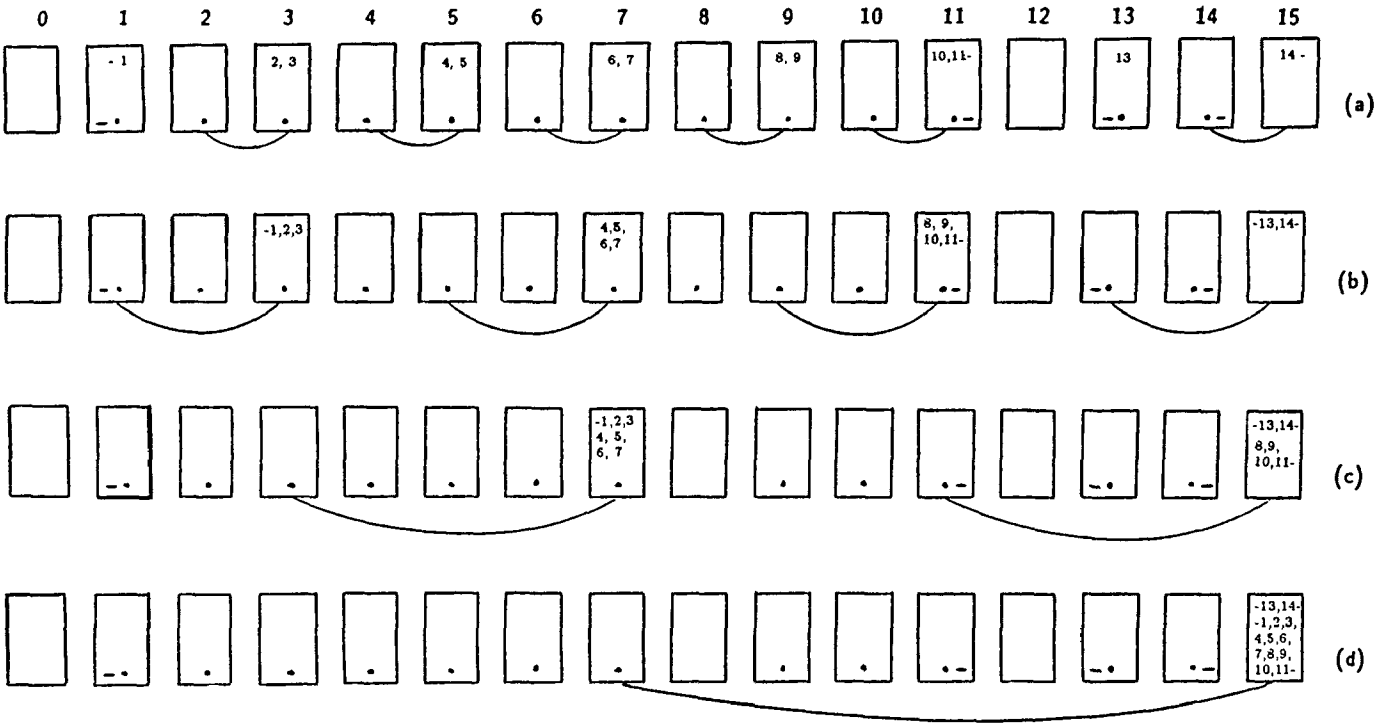


Fig.7. Stages of parallel shape coding of two disjoint horizontal (or vertical) segments on a two-dimensional array of PEs [25]. Dots represent presence of elementary segment detected by a PE, arcs show the transfers of messages containing feature chunks, mark '-' denotes a segment bound or tip: a), b), c) merging at the second, third and fourth levels of the parallel processing. PE#15 receives full set of the short segment in the third stage of parallel processing, and the long segment in the fifth stage (d)

have L=4, C=4 and R=1, so the basic step is proved. For the inductive step observe, that any new single region decrements L by 1 changing the inductive hypothesis (2) to $L-1=C+1-(R+1)$ and satisfying directly Eq.(2). •

Example 6. There are no tips in the OBJECT 4 in Fig.4. In the 2nd level of processing, the PEs #(15,11) and #(13,11) send their approximations to 3rd level PE #(15,11), and PEs #(15,13) and #(13,13) to 3rd level PE #(15,15). These 3rd level PEs detect the upper and lower object bounds, but they do not have the full sets of features. The 4th level PE#(15,15) detects the region and two bounds more: left and right. It has the full set of features satisfying Eq.(2), so it announces recognition of the circular object.

4. Involving Forks and Junctions

Tips are given the highest priority in merging to adjacent features, because tips are not counted by Eqs.(1, 2). Somewhat lower priorities should be assigned to curvature bounds, corners and horizontal or vertical nonhomogeneous segments when merging them with oblique segments when making the *primal sketch* when collecting patches or blobs using Eq.(1). When detecting regions from patches, the patches are expected to be reconstructed already from bounds, corners and curves. Patches should have a lower priority than regions when making the $2\frac{1}{2}D$ *sketch* using Eq.(2). One approach in processing shapes is to assume that the priorities of processing forks and junctions are lower than those for regions and patches, because forks and junctions form 3D objects from the $2\frac{1}{2}D$ *sketch*. Then any regions and patches are merged to forks and to junctions. Forks and junctions incur more

complexity. Shapes involving forks and junctions are also amenable to a rule checking a full collection of primitive components of 3D objects as shown below.

THEOREM 3. Parallel coding of shape involving forks and junctions is completed if the following terminations condition is satisfied:

$$L = J * 3 + F * 2 + C + 1 - R, \qquad (3)$$

where J is the mumber of junctions; F is the number of forks; L, C, R have the same meanings as in Eq.(2).

PROOF (inductive). We assume the truth of Eq.(2) for the basis step. Let Eq.(3) be the inductive hypothesis. Any single fork increments L by two giving $L + 2 = J * 3 + (F + 1) * 2 + C + 1 - R$ and reducing to Eq.(3), and any single new junction increments L by three giving $L + 3 = (J + 3) * 3 + F * 2 + C + 1 - R$ and reducing to Eq.(3). •

The lines L, bounds C, forks F, junctions J and regions R (i.e. delimited by closed curves) are counted by ancestors on each level of parallel processing. Because of a higher complexity and lowest priority of forks and junctions, Eq.(3) can be checked only at nodes detecting forks and junctions. Partial approximations of object fragments adjacent to the forks and junctions will be sent to these nodes.

THEOREM 4. Parallel coding of shape involving any elementary features detectable by the numbers I_o and I_w of window objects (Fig.1) is completed if:

$$L = \sum_{features_f} |features_f| * f + 1 - R \qquad (4)$$

where $|features_f|$ is the mumber of features of the order f. The *feature order* f is defined by the number I_o of window background parts minus one, i.e.:

$$f = I_o - 1 \qquad (5)$$

For example, $|features_f| = F$ for $f = 2$ is the number of forks, and $|features_f| = C$ for $f = 1$ is the number of corners or curve bounds.

PROOF (inductive). We assume the truth of Eq.(3) for the basis step (i.e., when having 0 features of orders $f > 2$). Let Eq.(4) be the inductive hypothesis. Assuming the presence of only the features of one selected order $f > 2$ and junctions J, forks F, corners of bounds C with possible regions, any new single f-order feature, $f > 2$, increments L by f yielding $L + f = (|features_f| * f + 1) + J * 3 + F * 2 + C + 1 - R$ and reducing to Eq.(4). The same is true for features of any order f. •

DEFINITION 1. An object is said to be recognized if the numbers L, C, F, J, $|features_f|$, R satisfy one of the Eqs.(1), (2), (3) or (4), and if each of its lines contains two endings, such as tip-tip, or tip-bound, or bound-bound, or tip-fork, or bound-junction, etc.

THEOREM 5. When shape coding of an object is terminated, the root of the parallel computation has a complete set of features (i.e., one of Eqs.(1), (2), (3) or (4) is satisfied at the root).

PROOF. In each step of parallel processing the number of branches of any computation tree decreases because of the merging processes. Because of different priorities, Eq.(1) is checked first to detect all 2D drawings and patches. If the drawings are part of regions or are adjacent to regions and planes, then Eq.(1) is not satisfied because it does not consider regions. If image regions are considered, then the regions may be a part of 3D objects or adjacent to them, and then Eq.(2) is not satisfied because it does not consider forks and junctions formed by regions or planes. Hence, Eq.(3) must be used, or even (4). Only two cases are possible: a) A node processing one object contains a partial set of features (e.g., the shape coding is not completed yet or an object is only partly located in the image plane), or b) a node is a root and has a complete set of features. •

THEOREM 6. Any feature or object merged and collected by a node in the last step of a completed parallel processing is continuous.

PROOF. If there is a full set of features collected by the node then all the features must make one entity. Any disjoint set of features (in the image plane) has its own

full set of features (satisfying one of Eqs. (1) to (4)). •

LEMMA 3. The complexity of the parallel shape coding is $O(log_2(object\_maximum\_length))$ parallel stages.

PROOF. When collecting (and merging) features of an object of the maximum length N, $2^{l-1} \leq N \leq 2^l$, each new step involves 2 times nodes less, where l is the level of parallel processing. •

5. Conclusions

The step of parallel shape coding entails collecting adjacent feature segments and merging them together according to partly known object chunks. In general, the parallel shape coding creates a topological graph of the absolute or relative positions of features and their particular parameters. The graph nodes are labeled by feature names, and arcs represent relations on the features. Thus, the feature names and their parameters are determined and localized through the object shape.

The approach follows the *criterion of simplicity:* a generalization of the Gestalt laws saying that the least information is more likely to be the best [18]. The least information becomes representation of the most regular shapes derived from the picture and assumes the names for the shapes. The least information must be on the symbolic level, because only symbolic information is independent of possible orientations of the pattern in the field of observation [24, 26]. The symbolic representations of features is shown to be invariant in the space of observation. The approach has the property of mapping an image fragment directly into words and phrases. Mathematical calculus for the proposed parallel shape recognition can be based on the concept of the rough sets [24, 12, 29, 30].

Template matching (see [18]) is a time consuming, rotation-dependent and ambiguous. Hong, Shneier, Hartley and Rosenfeld [8] merge in parallel small feature pieces into entities on a pyramid architecture, to speed up the recognition of image edges from matched templates. They exploit good continuation of adjacent window edges to remedy the ambiguity in interpretation of edges matched by single (separate) templates. The proposed parallel shape coding method is less ambiguous when compared to template matching (because feature names are detected in a deterministic, and not in a statistical way) and when compared to chain coding (because is resistant to variable thickness and discontinuities of objects). Parallel relaxation [17, 13] can improve the algorithms to cope with the problem of discontinuities in the analyzed figures.

Our symbolic and quantitative evaluation of window contents and merging of elementary features into wholes is also a remedy to the problem of ignoring feature continuity by pure clustering methods. For example, the Hough transform [4] in some cases may lose the track of good continuity of edge by treating collinear strips far away from each other as proximate segments. The good continuity is not lost, however, if a separate Hough transform is made for different window contents (extremely helpful when the number of background parts is 1).

Our termination conditions of shape coding is applicable not only to the 'block world', but also to objects with curved surfaces. The up-to-date experience with the sequential implementation of the shape detection method [24-26] prove the following: there is much less ambiguity in interpretation of elementary and complex features, when compared with template matching. The method is resistant to variable thickness and discontinuities of edges. Is not sensitive to a variety of disturbances. A window is analyzed only once. Particular parameters of features are determined with great precision. Underlying subroutines have already been tested on complicated patterns consisting of lines, arcs, corners and forks, and with very good results [24-26]. The termination conditions has been successfully applied for sequential shape coding, too.

References

[1] D.H. Ballard, C.M. Brown, *Computer Vision*, Prentice Hall, 1982.

[2] W.T. Beyer, *Recognition of topological invariant by iterative arrays*, Ph.D. thesis, MIT, 1969.

[3] P.R. Cohen, E. Feigenbaum, *The Handbook of Artificial Intelligence*, Addison - Wesley, 1982.

[4] R.O. Duda, P.E. Hart, "Use of the Hough Transformation to Detect lines and Curves in Pictures", *Communication of the ACM*, Vol. 15, No. 1, pp. 11-15, 1972.

[5] R.P. Grimaldi, *Discrete and Combinatorial Mathematics*, Addison-Wesley, 1989.

[6] D.S. Hischberg, A.K. Chandra, D.V. Sarwate, "Computing connected components on parallel computers", *Comm. ACM*, 22, 1979, pp461-464.

[7] T.H. Hong, M. Shneier, "Extracting Compact Objects Using Linked Pyramids", *IEEE Trans. on PAMI*, Vol. PAMI-6, No;.2, March 1984, pp229-237.

[8] T.H. Hong, M. Shneier, R.L. Hartley, A. Rosenfeld, "Using Pyramids to detect Good Continuation", *IEEE Trans. on Systems, Man and Cybernetics*, Vol. SMC-13, No.4, July / August 1983, pp631-635.

[9] K. Hwang, F.A. Briggs, *Computer architecture and parallel processing*, McGraw-Hill, 1984.

[10] G.F. Luger, W.A. Stubblefield, *Artificial Intelligence and the Design of Expert Systems*, Benjamin / Cummings, 1989.

[11] R.S. Michalski, J.G. Carbonell, T.M. Mitchell, *Machine Learning: An Artificial Intelligence Approach*, Morgan Kaufmann, 1986.

[12] Z. Pawlak, "Rough Classification". *International Journal of Man-Machine Studies*, 20, 1984, pp469-483.

[13] J.M. Prager, "Extracting and Labeling Boundary Segments in Natural Scenes", *IEEE Trans. PAMI*, v.2, No.1, pp16-27, 1980.

[14] E. Rich, *Artificial Intelligence*, McGraw-Hill, 1983.

[15] K.H. Rosen, *Discrete Mathematics*, Random House, 1988.

[16] A. Rosenfeld, "Some pyramid techniques for image segmentation", *NATO ASI series*, Vol.F25, in: *Pyramid Systems for Computer Vision*, Eds. V. Cantoni S. Levialdi, Springer-Verlag, 1986, pp261-271.

[17] A. Rosenfeld, R.A. Hummel, S.W. Zucker, "Scene Labeling by Relaxation Operations", *IEEE Trans. Syst., Man, Cybern.*, v.SMC-6, pp420-433, 1976.

[18] A. Rosenfeld, A.C. Kak, *Digital Picture Processing*, Acad. Press, 1982.

[19] M.O. Shneier, "Extracting Features from Images Using Pyramids", *IEEE Trans. Syst., Man, Cybern.*, v.SMC-12, No.4, pp569-572, 1982.

[20] Q.F. Stout, "Broadcasting in mesh-connected computers", *Proc. 1982 Conf. on Inform. Sci. and Systems*, pp85-90.

[21] Q.F. Stout, "Supporting divide-and conquer algorithms for image processing", *Journal of Parallel and Distributed Computing*, 4, 1987, pp.95-115.

[22] S. Tanimoto, *The Elements of Artificial Intelligence*, Computer Science Press, 1988.

[23] P.H. Winston, *Artificial Intelligence*, Addison-Wesley, 1984.

[24] Z.M. Wójcik, "Rough Approximation of Shapes in Pattern Recognition", *CVGIP*, 40, 228-249, 1987.

[25] Z.M. Wójcik, "A Natural Approach in Image Processing and Pattern Recognition: Rotating Neighborhood Technique, Self-Adapting Threshold, Segmentation and Shape Recognition", *Pattern Recognition*, v.18, no.5, 1985, pp299-326.

[26] Z.M. Wójcik, "An Approach to the Recognition of Contours and Line-Shaped Objects", *CVGIP*, 25, 1984, pp184-204.

[27] Z.M. Wójcik, A. Rosenfeld, "A Shape Coding Array", *Pattern Recognition Letters*, 4, 1986, pp57-59.

[28] Z.M. Wójcik, "A Parallel Shape Coding on SIMD Architecture", *Proc. IEEE Intern. Workshop on Tools for AI*, Fairfax, Oct. 1989.

[29] Z.M. Wójcik, "The Rough Grammar for Parallel Shape Coding, *Proc. ACM South Central Regional Conf.*, Nov. 1989, Tulsa.

[30] Z.M. Wójcik, B.E. Wójcik, "A Rough Grammar for a Linguistic Recognition of Image Patches", *Signal Processing*, 19, 1990, pp119-138.