

Peculiarities of Structural Analysis of Image Contours Under Various Orders of Scanning

V. Matsello, M. Schlesinger
Institute of Cybernetics
Ukrainian Academy of Sciences
Kiev, Ukraine

Abstract- The exact matching task for contour and formally defined set of contours is considered. The set of contours is defined by means of formal tools similar to the description of regular languages, but taking into account the cyclic structure of the contour when the starting point of contour tracing in general case is not defined. The algorithm is proposed for structural analysis of contours during line-by-line scanning of the image. Experimental results show high performance of the algorithm.

1. Introduction

It was a recognition of cartographic symbols of certain type that caused the present investigation. As far as the pictures under recognition are binary ones, it is natural that the pictures should be recognized using the information about their contours. On the one hand, the contour description contains the same complete information about a picture as the primary raster. On the other hand, the contour representation can be considered as a function that is defined on the one-dimensional cell complex [1]. It makes the related analysis much more simple than the analysis of raster representation, whose domain is a two-dimensional cell complex. In spite of this one-dimensionality of contours, their analysis can be very rarely reduced directly to the analysis of the strings, as it was made in [2]. This is because a string has an unclosed structure with two elements, which form the ends of the string, while contour has a cyclic structure with no such elements [3]. When a string is analyzed from left to right, there is a single element at every step of the analysis that has been already observed and that has a neighbor among the elements, which have not been observed yet. That is, there is a single point that serves as a border between observed part of the string and remaining part. As to contour analysis, there are two such border points even if it is possible to analyze contour's elements one-by-one in the order they occur during contour tracing. If such tracing is not possible, e.g. if contour should be analyzed during line-by-line scanning of the picture, the subset of contour's elements which are already examined may consist of many disjoint parts and the number of border points between observed and remaining elements can be large enough. All these peculiarities of contours complicate their structural analysis but do not make it hopeless. The basic concept used here for the statement and solution of structural analysis task is the set of allowable contours. This concept is defined by some formal tools

which are similar to formal grammars but taking into account the above mentioned distinctions between contours and strings. These tools, as well as the exact matching problem are defined in the next section. Two algorithms of the exact matching are described. The first of them is suitable if contour's elements are analyzed in the order they occur during the contour tracing. The second one is intended for the analysis of contours during line-by-line scanning of the image. In the section 3 the general form of the best matching problem is stated and the general form of its solution is described. Implementation of proposed ideas to the development of the program for recognition of hand written mapping symbols is described in section 4.

2. Exact Matching Problem and Related Algorithms

2.1. Formal Statement of the Problem

Let T be finite set. Let us assume that for every element $t \in T$ the next element t^+ and preceding element t^- are defined, such that for every $t \in T$ the following equalities hold

$$(t^+)^- = (t^-)^+ \text{ and } t^+ \neq t, t^- \neq t.$$

The set T is a *cycle* or a *group of cycles*. Let V be the set of signals and S be a finite set of states. A contour is defined as a function $v: T \rightarrow V$. A set of allowable contours is defined by the function $f: S \times V \times S \rightarrow \{0,1\}$. The contour $v: T \rightarrow V$ is allowable with the respect to the function f if there exists the function $s: T \rightarrow S$ such that for every $t \in T$ the condition $f(s(t^-), v(t), s(t)) = 1$ is satisfied. To solve the exact matching problem means to construct the algorithm that for every given contour $v: T \rightarrow V$ and given function $f: S \times V \times S \rightarrow \{0,1\}$ decides, whether v is allowable with respect to f . It means that exact matching consists in calculating of the value

$$F(v) = \exists s \{ \forall t [f(s(t^-), v(t), s(t))] \}. \quad (1)$$

2.2. Exact Matching of Strings

The predicate (1) is similar to those for decision whether a string belongs to regular language. Really, let V and S are terminal and nonterminal alphabets of some regular grammar G and $\sigma \in S$ be its axiom. In this case the function $f: S \times V \times S \rightarrow \{0,1\}$ corresponds to the set of production rules of type $s \rightarrow vs', s \in S, v \in V, s' \in S$, and the function $\phi: S \times V \rightarrow \{0,1\}$ corresponds to the set of production rules of type $s \rightarrow v, s \in S, v \in V$. The string $v = v(1), v(2), \dots, v(n)$ is considered to belong to the language of the grammar G if

$$F(v) = \exists s(0), s(2), \dots, s(n-1) \{ [\forall t = 1, 2, \dots, n-1 (f(s(t-1), v(t), s(t)))] \ \& \ \phi(s(n-1), v(n)) \} = 1, \quad (2)$$

with $s(0) = \sigma$.

It is known that the predicate (2) can be calculated by means of the following auxiliary functions $g_i: S \rightarrow \{0,1\}$, $i = 0,1,\dots,n-1$, $s \in S$, using the following expressions

$$\begin{aligned} g_0(s) &= 1, \text{ if } s = \sigma, \\ &0, \text{ if } s \neq \sigma; \\ g_i(s) &= \exists s' [g_{i-1}(s') \& f(s', v(i), s)], s \in S, i = 1, 2, \dots, n-1; \\ F(v) &= \exists s [g_{i-1}(s) \& \varphi(s, v(n))]. \end{aligned} \quad (3)$$

The total computational complexity of (3) is of order $k^2 \cdot n$, where k is the number of nonterminal symbols.

2.3. Exact Matching of Contours During Contour Tracing

In this section the algorithm for calculating of (1) will be described for the case when it is possible to trace the contour. It means that the contour is analyzed element-by-element so, that if the element $v(t)$ has been examined at some step then the element $v(t^+)$ should be examined at the next step. Let us consider the step when some current $t \in T$ is examined and the current signal $v(t)$ is observed. The cycle T , which t belongs to, will be called the current cycle.

Before the analysis of the signal $v(t)$ the algorithm has at its disposal the certain information about the part of contour that was already examined. This information is arranged as a function $g_{t^-}: S \times S \rightarrow \{0,1\}$, where t^- is the last element among the elements that are already examined. The value $g_{t^-}(s, s')$ determines, whether up-to-now examined part of contour is allowable under the condition that the start and final states associated with this part are s and s' correspondingly. At the step when the signal $v(t)$ is observed the algorithm constructs the function $g_t: S \times S \rightarrow \{0,1\}$, using the current value $v(t)$ and, in general, the function g_{t^-} according to the following rules.

a). If the current element t is the first element of the current cycle then

$$g_t(s, s') = f(s, v(t), s'). \quad (4)$$

b). If the current element t is the last element of the current cycle and the current cycle is not the last one, then

$$\text{if } \exists s, s' [g_{t^-}(s, s') \& f(s', v(t), s)] \quad (5)$$

then no action is fulfilled;

else decision is made that the contour is not allowable.

c). If the current element t is the last element of the current cycle and the current cycle is also the last in the group of cycles, then

$$\text{if } \exists s, s' [g_{t^-}(s, s') \& f(s', v(t), s)]$$

then the decision is made that the contour is allowable;

else decision is made that the contour is not allowable.

d). If the current element t is neither the first nor the last element of the current cycle then

$$g_t(s, s') = \exists s'' [g_{t^-}(s, s'') \& f(s'', v(t), s')]. \quad (6)$$

Computational complexity of the calculation is defined by computational complexity of (6) and is of order $k^3 \cdot n$. Such growth of complexity in comparison with the string matching is an inevitable expenditure for the dealing with the cyclic structure, which is more complex than a string. Nevertheless, under certain conditions the order of the complexity can be reduced.

Let for some t the function of two variables $g_t(s, s')$ can be represented by two functions of one variable such that

$$g_t(s, s') = g^b(s) \& g_t^e(s'). \quad (7)$$

If this condition is valid for some t then computational complexity of the rest of computations for the current cycle can be reduced from $k^3 \cdot n'$ to $k^2 \cdot n'$, where n' is the number of the rest of elements in the current cycle. Really, if (7) occurs, then

$$\begin{aligned} g_{t^+}(s, s') &= \exists s'' [g^b(s) \& g_t^e(s'') \& f(s'', v(t^+), s')] = \\ &= g^b(s) \& \{\exists s'' [g_t^e(s'') \& f(s'', v(t^+), s')]\}. \end{aligned} \quad (8)$$

Let denote

$$g_{t^+}^e(s') = \exists s'' [g_t^e(s'') \& f(s'', v(t^+), s')] \quad (9)$$

and therefore obtain

$$g_{t^+}(s, s') = g^b(s) \& g_{t^+}^e(s'). \quad (10)$$

It means that if for some t the condition (7) is valid then it is also valid for t^+ . Moreover, the first multiplier from (7) is not changed and the second multiplier must be calculated using (9). So, the complexity order is reduced in comparison with (6).

2.4. Exact Matching of Contours During Arbitrary Ordered Examination of Contour Elements

Let us consider the situation when the algorithm can observe signals $v(t)$ not during the contour tracing but in arbitrary order, for example during line-by-line scanning of the image.

Let at current step the algorithm observe the pair t and $v(t)$, where t is a contour element and $v(t)$ is the signal that corresponds to this element. Let us suppose that at this step the algorithm has already observed some set (possibly empty one) of contour signals. This set consists of some connected subsets of contour segments. We will refer these subsets as connected segments.

Let T' be one of observed connected contour segments that is not a complete cycle yet. Such segment has start element and final element. Let

t_b be the element that precedes the start element and t_e be the final element. Let $s(T')$ be the function $T' \setminus \{t_e\} \rightarrow S$. Let us denote

$$g_{t_b, t_e}(s(t_b), s(t_e)) = \exists s(T') [\& f(s(t^-), v(t), s(t))].$$

$t \in T$

The recognition algorithm must work in such a way that at every step it must keep in its memory the list of triples (t_b, t_e, g_{t_b, t_e}) , where each triple corresponds to the observed connected contour segment that still is not cycle. We will describe the recognition algorithm as the set of actions that must be performed at every step when the algorithm observes the next contour element (a pair $(t, v(t))$). Those actions depend on how this element is connected with elements that have already been observed at previous steps. Let us describe those actions.

a) *Joining to the end of the segment.* If in the list of triples there is a triple (t_b, t_e, g_{t_b, t_e}) such that $t_e = t^-$ and there is no such triple that $t_b = t$, then this triple is excluded from the list and new triple $(t_b, t, g_{t_b, t})$ is introduced, where $g_{t_b, t}$ is calculated as

$$g_{t_b, t}(s(t_b), s(t)) = \exists s(t_e) [g_{t_b, t_e}(s(t_b), s(t_e)) \& f(s(t_e), v(t), s(t))]. \quad (11)$$

b) *Joining to the beginning of the segment.* If in the list of triples there is a triple (t_b, t_e, g_{t_b, t_e}) such that $t_b = t$ then triple (t_b, t_e, g_{t_b, t_e}) is replaced by new triple (t^-, t_e, g_{t^-, t_e}) , where

$$g_{t^-, t_e}(s(t^-), s(t_e)) = \exists s(t) [f(s(t^-), v(t), s(t)) \& g_{t, t_e}(s(t), s(t_e))]. \quad (12)$$

c) *Merging of contour segments.* If in the list of triples there are a triple (t_b, t^-, g_{t_b, t^-}) and a triple (t, t_e, g_{t, t_e}) and those triples do not coincide, both of the triples are replaced by one new triple (t_b, t_e, g_{t_b, t_e}) , where

$$g_{t_b, t_e}(s(t_b), s(t_e)) = \exists s(t^-), s(t) [g_{t_b, t^-}(s(t_b), s(t^-)) \& \\ \& f(s(t^-), v(t), s(t)) \& \\ \& g_{t, t_e}(s(t), s(t_e))]. \quad (13)$$

d) *Completing the cycle.* If in the list of triples there is a triple (t, t^-, g_{t, t^-}) then we calculate the value

$$R = \exists s(t), s(t^-) [g_{t, t^-}(s(t), s(t^-)) \& f(s(t^-), v(t), s(t))]. \quad (14)$$

If $R = 0$ then the contour is not allowable.

If $R = 1$ and the element $(t, v(t))$ is the last element of the contour then decision is made that the contour is allowable.

If $R = 1$ and the element $(t, v(t))$ is not the last element of the contour (for example, if contour consists of several cycles) then the triple

(t, t^-, g_{t,t^-}) is excluded from the list and we must continue contour analysis.

e) *Starting new cycle.* If in the list of triples there are neither triple (t_b, t^-, g_{t_b,t^-}) nor triple (t, t_e, g_{t,t_e}) then the new triple $(t^-, t, g_{t^-,t})$ is introduced, where

$$g_{t^-,t}(s(t^-), s(t)) = f(s(t^-), v(t), s(t)). \quad (15)$$

If it is possible to arrange the storing of the list of triples in such a way that checking of presence of the triple in the list does not depend on the volumes of S and T then computational complexity of the algorithm is determined by the computational complexity of calculations of (11)-(15). In this case the complexity is of order $k^3 \cdot n$, that is the same as in the case of contour tracing. Moreover, in the above-mentioned algorithm we can also use the fact that if some function $g_{t,t'}(s, s')$ can be represented as $g_t(s) \& g_{t'}(s')$ then starting from this step the further analysis of the cycle will have the complexity of order k^2 instead of k^3 .

3. Best Matching Problem

There are several different definitions for the problem of evaluation of the correspondence between an object and the class of objects. We will consider three of them, propose the general formulation of the problem and discuss an algorithm for solving the problem.

The most widespread method consists in introduction of some metric over the set of objects and the task itself is to find such an object from the given set that is nearest to examined one in this metric. Such minimal distance estimates how precisely the given object matches with an object from the given set. It was shown in [4] that if Levenstein metric [5] is used, then the best matching problem is reduced to calculation of the function

$$F(v) = \min_{S, t \in T} f(s(t^-), v(t), s(t)), \quad (16)$$

where f is some function taking real values.

Another class of best matching problems uses ideas of fuzzy sets. In this case the task is reduced to calculation of the function

$$F(v) = \min_{S, t \in T} \max f(s(t^-), v(t), s(t)). \quad (17)$$

For the best matching problems of the third class it is necessary to calculate the distance from examined object to q nearest objects from the set [6].

Let us formulate the general definition of these problems. Let V and S be the finite set of signals and finite set of states (nonterminals), T be a cycle or a group of cycles, $v: T \rightarrow V$ be a contour and f be the

function which has the set $S \times V \times S$ as its domain and takes its values on the commutative semi-ring W [7]. Let the ring operations denoted by \oplus and \otimes are defined for W . All above listed types of best matching problem can be formulated as the calculation of the element from W in accordance with the formula

$$F(v) = \bigoplus_{s \in T} \otimes f(s(t^-), v(t), s(t)). \quad (18)$$

In (18) the sign \bigoplus_s means the "summation" over the set of all possible functions $T \rightarrow S$.

Expression (18) coincides with expression (1) if W is the set $\{0,1\}$, \oplus means disjunction and \otimes means conjunction.

It coincides with expression (16) if W is, for example, the set of integers, \oplus means *min* and \otimes means addition. If \oplus means *min* and \otimes means *max* it coincides with (17).

The third class of best matching problems can be reduced to (18) if W is the set of nondecreasing sequences each of them having length q . Let $x = (x_1, x_2, \dots, x_q)$ and $y = (y_1, y_2, \dots, y_q)$ be such two sequences. In this case the sequence $x \oplus y$ is defined as follows: it is necessary to make the concatenation of x and y , to order the resulting sequence as nondecreasing one and to take its first q elements. To obtain the sequence $x \otimes y$ it is necessary to calculate q^2 values of $x_i + y_j$, to order obtained set as nondecreasing sequence and to take its first q elements.

The algorithm for calculating of (18) for the case of arbitrary order scanning of contour elements has the same structure as it was described in section 2.4 for calculating of (1). Expressions (11)--(14) must be rewritten in generalized form as follows

For (11)

$$g_{t_b, t_e}(s(t_b), s(t)) = \bigoplus_{s(t_e)} [g_{t_b, t_e}(s(t_b), s(t_e)) \otimes f(s(t_e), v(t), s(t))]. \quad (19)$$

For (12)

$$g_{t^-, t_e}(s(t^-), s(t_e)) = \bigoplus_{s(t)} [f(s(t^-), v(t), s(t)) \otimes g_{t, t_e}(s(t), s(t_e))]. \quad (20)$$

For (13)

$$g_{t_b, t_e}(s(t_b), s(t_e)) = \bigoplus_{s(t^-)s(t)} \bigoplus [g_{t_b, t^-}(s(t_b), s(t^-)) \otimes f(s(t^-), v(t), s(t)) \otimes g_{t, t_e}(s(t), s(t_e))]. \quad (21)$$

For (14)

$$R = \bigoplus_{s(t)s(t^-)} \bigoplus [g_{t, t^-}(s(t), s(t^-)) \otimes f(s(t^-), v(t), s(t))]. \quad (22)$$

The condition of possibility to reduce the computational complexity is a generalization of the condition (7) and can be expressed as

$$g_{t,t'}(s,s') = g_t(s) \otimes g_{t'}(s').$$

4. Implementation

Some of proposed ideas were implemented for the development of the program for recognition of mapping symbols. In the program contours were set by coordinates of ends of contour segments $v(t) = \{x(t), y(t)\}$. Unlike the function $f: S \times V \times S \rightarrow \{0,1\}$ defined in section 2.1 for definition of the set of contours the couple of functions $f_1: S \times V$ and $f_2: S \times S$ were used. The function f_1 defines what states from S are allowable in every image pixel. The function f_2 describes the order in which states can follow each other during contour tracing. When constructing this function, it is possible to take into account the most expectable image distortions.

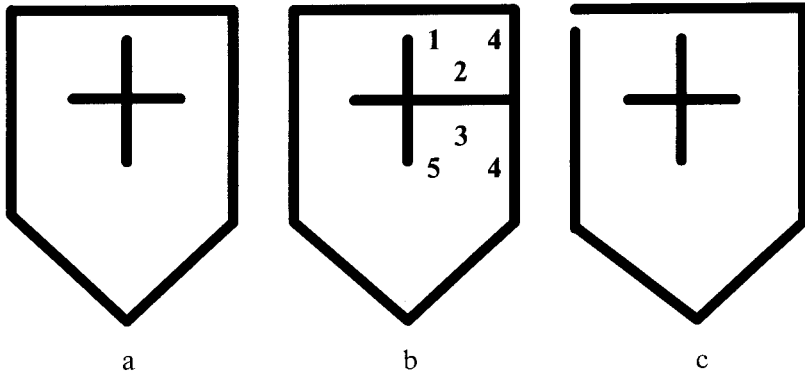


Fig. 1. Mapping symbol “hospital” a) without distortions, b) two lines are stucked together, c) break in up-left corner.

In the Fig.1.b the numbers 1-5 represent states associated with some contour segments. To allow both “normal” symbols (Fig.1.a) and sticking together of two lines (their contour segments correspond to states 2,3,4) the function f_2 must take the value 1 for the following pairs of states (2,2),(2,3),(2,4),(3,3),(4,4),(4,3).

Similar actions with some additional efforts can make allowable symbols with contour breaks (Fig.1.c).

The experimental program provides user with interactive tool for constructing functions f_1 and f_2 . Using sample image user should draw the “skeleton” of the symbol consisting of line segments and arcs and define maximum width of symbol lines. Then program automatically assigns corresponding states to contour segments and constructs the function f_1 . In such a way the image areas that correspond to every state are defined. Then the program displays every overlapping pair of areas and user should define

the order in which states associated with each area can follow each other during contour tracing. At this stage possible image distortions can be taken into account. Following user's orders the program constructs the function f_2 .

The program was tested on IBM PC 486-66 computer on several images with different hand drawing symbols. For the image containing 297 symbols of 6 types the recognition time of symbols of one type was from 0.05 sec. to 0.1 sec. depending of the quantity of symbols of each type.

5. Conclusion

The method for definition of sets of contours that correspond to graphical symbols is proposed. It provides the possibility to analyze images with the most frequent types of symbol distortions. The fast algorithm for structural analysis of contours during line-by-line scanning of the image is described. It allows efficient program realization.

References

1. V.A.Kovalevsky, "Topological Foundations of Shape Analysis", Proc. of the Workshop "Shape in Picture", September 7-11, 1992, Driebergen. The Netherlands.
2. K.S.Fu, "Syntactic Methods in Pattern Recognition", Academic Press, New York and London, 1974.
3. M.Hospital, H.Yamada, T.Kasvand, S.Umeyama, "3D Curve Based Matching Method Using Dynamic Programming", Proc. of the First International Conference on Computer Vision, June 8-11, 1987, London, England, pp.728-732.
4. M.I.Schlesinger, "Systeme von Functionsoperationen angewendet auf eine Aufgabe der besten Uebereinstimmung", //ISSN 0863-0798/Wissenschaftliche Beitrage zur Informatik - Fakultat Informatik TU Dresden/7(1994)Heft 3.-S.62-79.
5. В.И.Левенштейн, "Двоичные коды с исправлением выпадений, вставок и замещений символов"//Докл. АН СССР, 1965, 163, №4, с.840-850.
6. М.И.Шлезингер, "Математические средства обработки изображений", Киев, Наукова Думка, 1989.
7. A.V.Aho, J.E.Hopcroft, J.D.Ullman, "The Design and Analysis of Computer Algorithms", Addison-Wesley, Reading, Mass., 1975.