REGION REPRESENTATION USING ENHANCED DISCRETE CYLINDRICAL ALGEBRAIC DECOMPOSITION (EDCAD) TO PRESERVE THE SHAPE AND SIZE OF THE CONNECTED REGIONS IN BINARY IMAGES

B. Rajesh Kanna
St. Joseph's College of Engineering, Chennai, India,
rajesh@stjosephs.ac.in

C. Aravindan
S.S.N College of Engineering, Chennai, India,
AravindanC@ssn.edu.in

K. Kannan
SASTRA University, Thanjavur, India,
kkannan@maths.sastra.edu

M. Krishna Priya
St. Joseph's College of Engineering, India, Chennai,
krishnapriyamurugan@gmail.com

Abstract

In this paper we proposed a syntactic approach to represent any connected region taken from the binary digital image. The proposed method is an enhancement of DCAD algorithm and it provides alphanumeric string to represent the connected region, which preserves the shape and size. In enhanced DCAD we introduced two variables to differentiate the left and right bends to avoid the shape anomaly and we also computed the cylinder width and height relative to Region Of Interest's (ROI) width and height to preserve the dimension during the reconstruction of ROI. To evaluate our method we have created a database of connected regions in binary format and the regenerated connected region from our resultant alphanumeric string is compared with the actual one and found to be similar. The purpose of this EDCAD is to overcome the anomalies of size and orientation found in the original DCAD. This can be potentially used in applications like pattern recognition, pattern matching and retrieval, machine learning etc.

Keywords

Region representation, Connected regions, Decomposition and Shape and size.

1. Introductions and Motivation

More and more images have been generated in digital form around the world and there is a growing need to find images from large collection of databases. In order to find an image efficiently from the database, the image has to be represented by certain features. The region representation plays an important role in systems for object recognition and classification, shape matching and retrieval, machine learning and in image synthesis for graphics applications. For example in a trademark registry application [1], the new trademark can be ensured to be distinct from the existing marks by searching the database.

Dengsheng Zhang et.al [2] has classified the shape representation into two categories: boundary-based (also called as contour-based) methods and region-based methods. The following are some of the contour-based methods: Chain codes [3], Polygonal approximation [4] [5], Scale space method [6], Shape invariants [7] [8] [9] [10], etc. The disadvantage of contour-based method is that they can be applied only to simply connected region and not to multiply connected region. This limitation is can be overcome by using region-based methods. The following are some of the region-based methods: Algebraic moment invariants [11] [12], Orthogonal moments [13], Fourier descriptors [14], Grid based method [15], etc. The region-based method deals with connected regions. But the region-based methods are highly complex and it requires more memory space as image size increases. EDCAD takes the advantages of both contour-based and region-based methods.

EDCAD decomposes the connected regions using DCAD and represent each decomposed binary component using syntactic method (a contour-based method). Usually syntactic method uses predefined primitives to describe the orientation of the connected region but we have included dynamic primitive in EDCAD, which describes both the orientation and size of the connected region. The rest of the paper is organized as follows. The Section 2 reviews connected regions, CAD, and DCAD. In Section 3, we described the shape and size anomalies of the existing DCAD. The steps undertaken to overcome the problems of existing DCAD and the algorithms to implement EDCAD are discussed in Section 4. The Section 5 shows the experimental work and the discussion of the result. We conclude the paper in Section 6 by discussing possible extension of the work.

2. Background

Definition 1: (Binary image, Binary component [16])
A binary image D is a subset of the digital plane given by :

$$Z^2 : D = \{(i, j, g) \mid 1 \le i \le N, 1 \le j \le M, g \in \{0,1\}\}$$

Each element $x \in D$ is called pixel; gx denotes its intensity; the pair (ix, jx) indicates its position. A binary component of D is a subset of D whose pixel-values are 1.

Definition 2 : (Connected, Region [17])
A non-empty subset of Rj is called connected, if it is connected in a topological sense with respect to the topology induced on Rj by the Euclidean metric. We call a connected subset a region.

Definition 3 : (Jordan curve [18])
A Jordan curve is a closed curve C that does not intersect itself.

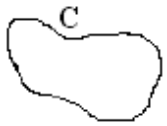

Fig 1 - Jordan Curve

Definition 4 : (Simply connected region [19])
A connected region R of a plane is called simply connected if and only if for every Jordan curve $J \subset R$ and every point $q \in R \backslash J$ in the bounded region determined by J, q belongs to R.

Definition 5 : (Multiply connected region [19])
If a connected region R of the plane is not simply connected then it is called multiply connected.

Definition 6 : (Decomposition, Cell [17])
A decomposition U of $R \subseteq R^j$ is a finite set of disjoint regions such that the union of U results in R. The elements of U are called cells.

Definition 7 : (Cylindrical decomposition, Cylindrical algebraic decomposition [17])
A decomposition Dj of Rj is called cylindric, if $0 \leq j \leq 1$ or if $j > 1$ and Dj can be partitioned into stacks over cells of a cylindrical decomposition of Rj-1, the induced cylindrical decomposition of Dj. A decomposition that is cylindric and algebraic is called Cylindrical Algebraic Decomposition (CAD).

Definition 8 : (Discrete Cylindrical Algebraic Decomposition (DCAD) [20])
The construction of CAD for analysis of points in the 2D discrete space is named as discrete CAD (DCAD).

Definition 9: (Edges in DCAD [16])
In DCAD, the internal edges of a connected component, $A \subseteq D$, plays the role of algebraic curves. The internal edges are defined as
Edge $(A)=\{x \in A: \exists y \in C_1^{(4)}(x)$ such that $g_y = 0\}$, where $C_1^{(4)}(x)$ is a 4-connected circle with radius 1 centered in x..
An edge of a component is said to be

Horizontal iff it is a sequence of type

(…. (ix, jx), (ix, jx+1)… (ix, jx+n)…..);

Bend iff it is a sequence of type

((ix , jx -1),(ix , jx),(ix-1 , jx),……..,(ix-n , jx), (ix-n , jx+1)) for left bend or ((ix, jx +1),(ix , jx), (ix-1, jx),….., (ix-n, jx),(ix-n, jx-1)) for right bend;

Open iff it is a sequence of type ((ix , jx +1), (ix, jx),(ix-1, jx),…(ix-n, jx),(ix-n, jx+1));
Close iff it is a sequence of type ((ix, jx -1), (ix, jx),(ix-1, jx),…(ix-n, jx),(ix-n, jx-1)).

Definition 9 : (Cylinders in DCAD [16], [17])
1) The cylinders are digital straight lines represented by a sequence of connected and vertically stacked pixels (…, (i_x, j_x), (i_x+1, j_x), (i_x+2, j_x), ., (i_x+n, j_x), …).
2) If S is a region in R^j, then:
cyl (S) := S × R denotes the cylinder over S.
In the DCAD, vertical straight lines, tangent to concavity, convexity, and bend points of borders of regular connected components are detected. This performs a partition of the image into cylinders. [21]. Hence the digital intersection $T \cap E$ of a cylinder T with an edge E of a connected component is a sequence of $n \geq 1$ connected pixels. The intersection is said to be

horizontal (p) if E is a horizontal edge (n==1);

opening (a) if E is an open edge;

closing (b) if E is a close edge;

bend (s) if E is a bend edge;

3. Shape and Size Anomalies of Existing DCAD

The discrete CAD [16] provides string representation of the ROI. The string is formed from the predefined five characters a, b, p, s, #. Each character represents a specific shape. The details about each character are given in the Definitions 7 and 8.

The representation of connected region using the combinations of those five characters gives insufficient information to regenerate the connected region form that string. 5

This issue is described in detail below. The existing DCAD does not address the following issues:

Shape Anomaly

The string representation of DCAD does not preserve the orientation of the ROI. Because the same character‘s’ is used to represent both the left and right bends as shown in Definitions 7 and 8. This leads to generation of different geometry for the same string during reconstruction, which is an anomaly.

For example in figure 2,

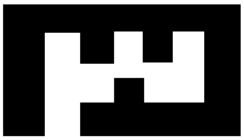

Fig 2 - Actual ROI

The string representation for the above figure is a#sp#asp#pssp#pssp#ppsp#pbp#b#.

The possible cases of ROIs reconstructed from the above string are shown in figure 3.

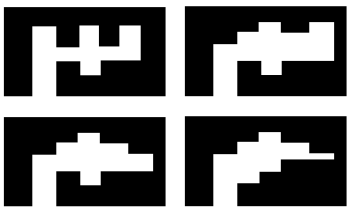

Fig 3 - Reconstructed ROI-Possible Cases

Lack of Size Information

Similarly the existing DCAD does not have any primitive to store the dimension information. So it is impossible to regenerate the ROI in its actual dimension from the available string, Thus the existing DCAD does not convey the shape and size information correctly through its string representation.

Proposed EDCAD

From the Section 3, it is clear that it is impossible to regenerate the ROI with existing DCAD's string representation. The Figure 3 shows that the character's' can be interpreted wrongly for both left and right bends. As the orientation of the bend changes the shape of the ROI also changes and it is also impossible to regenerate the actual ROI without knowing its dimension. To avoid these issues we have added few more definitions and corrections in EDCAD.

Shape Anomaly

To avoid the shape anomaly the Definitions 7 and 8 is slightly renamed. In existing DCAD, the character's' denotes both the left and right bends. This ambiguity is avoided by naming the left and right bends independently with the characters 'l' and 'r' respectively.

An edge of a component is said to be

Left bend (l) iff it is a sequence of type
((ix , jx -1),(ix , jx),(ix-1 , jx),……,(ix-n , jx), (ix-n , jx+1))

Right bend (r) iff it is a sequence of type
((ix , jx +1),(ix , jx),(ix-1 , jx),…..,(ix-n , jx), (ix-n , jx-1))

So the configuration of the edges of the binary component will be as shown in the figure 4.

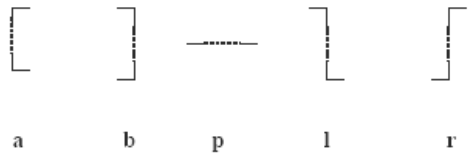

Fig 4 - Configuration of the Edges of Binary Component

Preservation of Dimension on Reconstruction

To regenerate the ROI in its actual dimension, the string representation of it must include the primitives for describing the dimension along with characters to describe the orientation. This is achieved by representing the ROI in alphanumeric format, where the numeric terms in it indicates the dimension and the characters in it indicate the orientation of the ROI. For representing the ROI in alphanumeric format, we have introduced Xcylinders and Ycylinders to store the dimension as the dynamic primitives.

The Xcylinders and Ycylinders are computed to find the width and height of the edges of ROI. These dynamic primitives are included in places next to the occurrences of the characters in the alphanumeric string. To represent the dimension in real world metric, it is given in relative proportions.

Definition 10: (Xcylinders, Ycylinders)

Xcylinders are digital straight lines represented by a sequence of connected and horizontally stacked pixels (…,(i_x, j_x), (i_x, j_x+1), (i_x, j_x+2),…., (i_x, j_x+n), …).

Ycylinders are digital straight lines represented by a sequence of connected and vertically stacked pixels (…, (i_x, j_x), (i_x+1, j_x), (i_x+2, j_x),…., (i_x+n, j_x), …).

The width of the ROI is measured by calculating the absolute difference between the values in x-axis (X_{min}, X_{max}). Xmin is the first occurrence of the point where the color of the ROI changes from white to black. Xmax is the last occurrence of the point where the color of the ROI changes from black to white, when scanning the ROI from left to right.

$$width = | X_{max} - X_{min} |$$

Similarly the height of the ROI is measured by calculating the absolute difference between the values in y-axis (Y_{min}, Y_{max}). Ymin is the first occurrence of the point where the color of the ROI changes from white to black. Ymax is the last occurrence of the point where the color of the ROI changes from black to white, while scanning the ROI from top to bottom.

$$height = | Y_{max} - Y_{min} |$$

The Ycylinders are formed on the ROI, when any of the edges a, b, l, and r occurs while scanning the ROI form left to right and the Xcylinders are formed on the ROI, when any of the edges a, b, l, and r within a particular Ycylinder intersects with it, while scanning the ROI form top to bottom. The width of the two consecutive Ycylinders is calculated by taking the absolute difference between the x-axis values where the two successive Ycylinders are formed. The width of the two consecutive Ycylinders is given by :

$$Ywidth = \{| i – j |, i, j \text{ are x-axis values}\}$$

The height of the two consecutive Xcylinders is calculated by taking the absolute difference between the y-axis values where the two successive Xcylinders are formed.

Xheight = {[i – j], i,j are y-axis values}

The relative proportion of Ywidth with that of width of the ROI is the width of all edges formed within that particular Ycylinders for which Ywidth is calculated and it is given by:

Width of the edge = Ywidth / width

The relative proportion of Xheight with that of height of the ROI is the height of the edge formed within that particular Xcylinders for which Xheight is calculated and it is given by:

Height of the edge = Xheight / height

Algorithm for finding the intersecting points of the cylinder with Image

function IntersectTest (Image)
returns intersecting pixel points of the entire image with the cylinder.
Inputs:
Image, Preprocessed binary image
Local variables:
width, width of the Image
height, height of the Image
black, foreground color of the Image
white, background color of the Image
x=1,y=1
Foreach x in width **do**
Foreach y in height **do**
Case 1:
colour of y is white & colour of y+1 is black
Store this y in the array
Case 2:
colour of y is black & colour of y+1 is white
Store this y in the array
Endfor
Endfor
endfunction

Algorithm for finding the alphanumeric representation of the ROI

function alphanumericRepresentation (intersec[], intersecno, y)
returns alphanumeric value for the given binary image
Inputs :
Intersec[], intersecting pixel points generated from IntersectTest function
intersecno, number of intersecting pixel points at y
y, number of cylinders
Local variable : i=0, j=0, k=1
Foreach i in y **do**
Foreach j in intersecno **do**
If intersec at (i,j) = intersec at (k,j) **And** color at (i-1,j) & color at (k,j) are black **Then**
Append ‘p’ in the character sequence
Else If intersec at (i,j) = intersec at (k,j) **And** color at (i-1,j) is white & color at (k,j) is black **Then**
Append ‘a’ along with (i,j) in the character sequence
Else If intersec at (i,j) = intersec at (k,j) **And** color at (i-1,j) is black & color at (k,j) is white **Then**
Append ‘b’ along with (i,j) in the character sequence
Else If intersec at (i,j) != intersec at (k,j) **And** j at i < j at k **Then**
Append ‘l’ along with (i,j) in the character sequence
Else If intersec at (i,j) != intersec at (k,j) **And** j at i > j at k **Then**
Append ‘r’ along with (i,j) in the character sequence
Endif
Endfor
Endfor
Return alphanumeric value
Endfunction

Algorithm for regenerating the object from the alphanumeric representation
function regenerate (chararray)
returns image generated form the alphanumeric value
Inputs:
chararry, contains the sequence of alphanumeric characters.
Local variables:
i=0, j=0, y1=0, y2=0, x=0
while i < length of chararray **do**
while j < index positon character ‘#’ in chararray **do**
If Character at i in the chararray is ‘a’ or ‘b’ or ‘l’ or ‘r’ **Then**
y1= substring of chararray from the index position of the character ‘(‘ to the index position of the character ‘,’
y2= substring of chararray from the index position of the character ‘,‘ to the index position of the character ‘)’
Draw line from y1 to y2
i= index position of the character ‘)’
Else If character at i in the chararray is ‘p’ **Then**
i=i+1
Else
x=substring of chararray from the index position of the character ‘(‘ to the index position of the character ‘)’
Draw line from pixel point at y1 & y2 till x
Endif
Endwhile
Endwhile
Endfunction

Worked Examples

We have worked with the images taken from custom shape tools (of fixed size 50 x 50 px) from the Photoshop version 6.0 and the EDCAD algorithm is implemented in java. Some of those images and its alphanumeric representations are given below.

Original Image	Alphanumeric Representation
■	a(19,28)9#b(19,28)#
□	a(8,23)4#p(23)a(11,19)p(8)8#p(23)b(11,19)p(8)4#b(8,23)#
▲	p(28)2#p(28)r(24,26)1#p(28)r(22,24)1#p(28)r(20,22)2# p(28)l(20,22)1#p(28)l(22,24)1#p(28)l(24,26)1#p(28)#
◆	l(25,26)r(23,24)2#l(26,27)r(22,23)1#l(27,28)r(21,22)2# r(27,28)l(21,22)1#r(26,27)l(22,23)1#r(25,26)l(23,24)#

Table 1: Example images taken and its alphanumeric representation

4. Result and Discussion

The generated alphanumeric representation is said to be accurate only if the image reproduced from that representation is same as that of the original one. The table below shows the alphanumeric representation and the image regenerated from it. When the regenerated image is given to the algorithm producing the alphanumeric representation, the one similar to the representation given in table 2 is got.

Alphanumeric Representation	Regenerated Image
p(29)a(20,22)p(20)2‡p(29)b(22,27)p(20)2‡r(27,29) l(20,21)1‡p(27)l(21,22)1‡l(27,29)r(20,22)2‡p(29) a(22,27)p(20)1‡p(29)b(20,22)p(20)‡	
l(26,28)r(21,23)2‡p(0)b(0,21)a(26,30)1‡l(30,31)l(26,28) r(21,23)r(18,19)1‡p(31)l(28,30)r(19,21)p(18)6‡p(31)r(28,30) l(19,21)p(18)1‡r(30,31)r(26,28)l(21,23)l(18,19)1‡b(26,30)1‡ p(0)b(0,21)p(0)p(0)p(0)p(0)‡	

regenerated image

5. Conclusion and Future Enhancements

Thus the region representation using EDCAD provides an opportunity to preserve the shape and size of the connected region in its representation. The alphabets in the alphanumeric representation describe the shape of the connected region and the numeric terms in it describes its size. However the length of the alphanumeric representation is long depending upon the size of the image, this can be simplified in future. This algorithm does not works well for the image with boundaries disturbed with noise, distortions etc. The EDCAD is focused on the image with single view. This can be extended to multiple views of the image in future.

References

1. Eakins JP (1994) Retrieval of trade mark images by shape feature. In: Proceedings first international conference on electronic library and visual information system research, de Montfort University, Milton Keynes, UK, pp 101–109.
2. D.Zhang, G.Lu, Review of shape representation and description techniques, Pattern Recognition 37 (2004) 1-19.
3. H. Freeman, On the encoding of arbitrary geometric configurations, IRE Trans. Electron. Comput. EC-10 (1961) 260–268.
4. W.I. Groskey, R. Mehrotra, Index-based object recognition in pictorial data management, Comput. Vision Graphics Image Process. 52 (1990) 416–436.
5. W.I. Groskey, P. Neo, R. Mehrotra, A pictorial index mechanism for model-based matching, Data Knowledge Eng. 8 (1992) 309–327.
6. G. Dudek, J.K. Tsotsos, Shape representation and recognition from multiscale curvature, Comput. Vision Image Understanding 68 (2) (1997) 170–189.
7. S.Z. Li, Shape matching based on invariants, in: O. Omidvar (Ed.), Shape Analysis, Progress in Neural Networks, Vol. 6, Ablex, Norwood, NJ, 1999, pp. 203–228.
8. C.-L. Huang, D.-H. Huang, A content-based image retrieval system, Image Vision Comput. 16 (1998) 149–163.
9. D.M. Squire, T.M. Caelli, Invariance signature: characterizing contours by their departures from invariance, Comput. Vision Image Understanding 77 (2000) 284–316.
10. M. Sonka, V. Hlavac, R. Boyle, Image Processing, Analysis and Machine Vision, Chapman & Hall, London, UK, NJ, 1993, pp. 193–242.
11. G. Taubin, D.B. Cooper, Recognition and positioning of rigid objects using algebraic moment invariants, SPIE Conference on Geometric Methods in Computer Vision, Vol. 1570, University of Florida, Florida, USA 1991, pp. 175–186.
12. G. Taubin, D.B. Cooper, Object recognition based on moment (or Algebraic, in: J. Mundy, A. Zisserman (Eds.), Geometric Invariance in Computer Vision, MIT Press, Cambridge, MA, 1992, pp. 375–397.
13. M.R. Teague, Image analysis via the general theory of moments, J. Opt. Soc. Am. 70 (8) (1980) 920–930.
14. D.S. Zhang, G. Lu, Generic Fourier descriptor for shape-based image retrieval, in: Proceedings of IEEE International Conference on Multimedia

and Expo (ICME2002), Vol. 1, Lausanne, Switzerland, August 26–29, 2002, pp. 425–428.
15. G.J. Lu, A. Sajjanhar, Region-based shape representation and similarity measure suitable for content-based image retrieval, Multimedia Syst. 7 (2) (1999) 165–174.
16. V.D.Gesu, C.Valenti, Two-view cylindrical decomposition of binary images, Linear Algebra and its Applications 339 (15) (2001) 205-219.
17. Andreas Seidl, Cylindrical decomposition under application-oriented paradigms, Doctoral Dissertation, Faculty of Mathematics and Informatics, University Passau.
18. M.J. Strauss, G.L.Brandley, K.J.Smith, Calculus, 3rd edition, Pearson Education, 2007, Ch.5 & 13.
19. U.H. Karimov, D.Repovs, M.Zeljko, On unions and intersections of simply connected planar sets, Monatshefte fur Mathematik 145 (2005) 239-245.
20. V.Di. Gesu, R.Renda, An algorithm to analyse connected components of binary images. Res.Informatics 4 (1991) 87-93.
21. F.Chiavetta, V.D. Gesu, Digital connectedness via connectivity graph, in: Proceedings of the 11th IAPR International conference on Image , speech, and signal Analysis, Vol.3., 1992, pp.646-649.