Representing 2D Digital Objects

Vito Di Gesù and Cesare Valenti

Dipartimento di Matematica ed Applicazioni, University of Palermo
Via Archirafi 34 - 90123 Palermo, Italy
{digesu,cvalenti}@math.unipa.it

Abstract. The paper describes the combination a multi-views approach to represent connected components of $2D$ binary images. The approach is based on the *Object Connectivity Graph* (OCG), which is a sub-graph of the connectivity graph generated by the *Discrete Cylindrical Algebraic Decomposition*($DCAD$) performed in the $2D$ discrete space. This construction allows us to find the number of connected components, to determine their connectivity degree, and to solve visibility problem. We show that the CAD construction, when performed on two orthogonal views, supply information to avoid ambiguities in the interpretation of each image component. The implementation of the algorithm is outlined and the computational complexities is given.

Keywords: shape representation, shape decomposition, shape description, digital topology

1 Introduction

The combination of multiple views of the same object can be used to improve its reconstruction. In principle, it is possible to recover the whole object information from partial ones; the number of views that are necessary to fully recover an object-representation depends also on the feature space. For examples in [1] two views are considered to reconstruct convex polyminoes; computerized tomography [2] is another example of reconstruction from projection. In [3] the symmetry transform, performed on multiple views, is used for object representation and classification.

The paper describes how to combine the information of two orthogonal views to represent the connected components in $2D$ binary images. The combination method uses the *Connectivity Graph* (CG) as it is derived from the *Cylindrical Algebraic Decomposition* (CAD)introduced by Collins [4] for the decomposition of the Euclidean space, E^d. In [5] the CAD construction has been extended to the analysis of components of a set of points of a $2D$ discrete space, $D \subseteq N^2$, where N is the set of natural numbers and it has been applied for the solution of geometry problems of the first and the second order [6,7], to computer aided design, and shape analysis.

The CAD-algorithm performs a cellular decomposition [8,9] of the projective plane in open ball, also named i-cells, with $i \geq 0$. For example in E^2 a 0-cell is a point, 1-cell is an open arc, and 2-cell is an open region. The cellular

G. Borgefors, I. Nyström, and G. Sanniti di Baja (Eds.): DGCI 2000, LNCS 1953, pp. 337–347, 2000.

decomposition of E^2 may be defined as a nested sequence $X^0 \subset X^1 \subset X^2 \subset E^2$ of closed subspaces, such that X^0 consists of finitely many 0-cells, X^1 consists of finitely many disjoint 1-cells, X^2 consists of finitely many disjoint 2-cells.

Let us describe the classical CAD-algorithm with an example. Let C be an algebraic curve in the real projective plane (see Fig.1a):

$$y^4 - 2xy^3 - x^2y^2 + y^2 + 2x^3y + x^2 - 1 = 0$$

then a cellular decomposition can be performed as in Fig.1b, by intersecting C with linear varieties. The decomposition is arbitrary, however after that it is performed the i-cells are determined, and they may be used to study C, and its connected components ($\leq \binom{n}{2} - 1$, where n is the degree of C). Two distinct cells

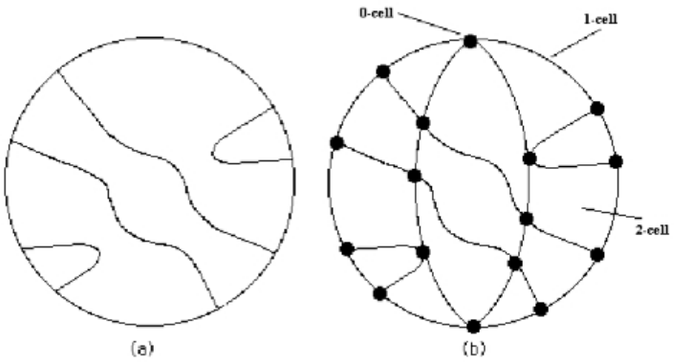

Fig. 1. a) Algebraic curve C in the projective real plane; b) The *Cylindrical Algebraic Decomposition*

are adjacent if they touch each other. Clearly adjacency is a symmetric relation. Therefore the CG nodes are cells, and the arcs represent the adjacency between cells.

The extension of the CAD decomposition to a digital spaces is named *Discrete CAD* ($DCAD$) [5]. Its definition is straightforward by providing the proper definitions of $i - cell$, and connectivity relation on D [10,11].

The $DCAD$ provides unambiguous topological information (connected components, holes); however, the structural description of a given component could be ambiguous. This ambiguity can be avoided by using two or more views. In the paper we show that two views are sufficient to avoid shape ambiguities that are on the component borders. An algorithm for combining two orthogonal views of the $DCAD$ construction is also given and its computational complexity outlined.

The paper is organized as follows. Section 2 describes the $DCAD$-algorithm. The algorithm to combine the CG is described in Section 3. Section 4 reports results and implementation notes. Final remarks are given in Section5.

2 The Discrete CAD Algorithm

In the following the internal borders of connected components in E^2 will play the role of the algebraic curves. In this case the CAD-algorithm consists in the partition of the space by means of parallel straight lines that intersect the borders of connected components. The i-cells, generated by the decomposition, are points (p), bend-points (s), open lines representing borders (b), open lines representing cylinders (c) and open $2D$ regions (r).

Starting from this decomposition a labeled not oriented connectivity graph (CG) is derived, nodes of which are of type p, s, b, c, and r. The arcs of a CG connect only nodes of different type that are adjacent. Fig.s 2a,b show an example of CAD, and the corresponding CG in E^2. An ordered pair of integer number, (L, R), is assigned to each node of the CG; where L and R are named the left and right labels respectively. This labeling allows us to identify univocally nodes of type c, r, and b, while nodes of type p and s have the same kind of pair, as stated below.

R/L	$0dd$	$Even$
Odd	c	r
$Even$	p,s	b

This labeling rule is determined by scanning the plane left to right (bottom to up), and increasing by 1 L (R) each time a new element of the decomposition is intersected (initially $L = R = 0$). For example the region r1 in Fig.2a is labeled $(0, 1)$, the line $c2$ is labeled $(1, 3)$ the line $b1$ is labeled $(2, 2)$ and the point $p2$ is labeled $(3, 2)$, see Fig.2b. In E^2 an infinity number of CAD's can be performed; in fact, it is possible to consider all possible directions of straight lines. Here, only straight lines, parallel to the $Y(X)$-axis that intersect concavity, convexity, and bend-points are selected.

The $DCAD$-decomposition is obtained by considering as cylinders digital straight paths in D that are parallel to the $Y(X)$-axis. Moreover, the cylinders selected cross the internal borders of the components in D, where one of the configurations of pixels shown in Fig.3 is present. These configurations correspond to concavities, convexities, and bends of digital borders.

The $DCAD$ construction may generate *empty regions* between two adjacent cylinders. In the algorithm these regions are treated as *virtual regions*, and their labeling is determined by considering the labels of adjacent cylinders. Each of these configurations became a single vertex in the construction of the CG. The labeling of CG is determined as in the continuous space, by scanning D left-right and bottom-up. The arcs represent the adjacency relation among points, paths and regions.

Note that the 4-adjacency holds between points, paths, and internal regions; while 8-adjacency holds between points and external regions.

2.1 Properties of the CG

The CG allows to retrieve the connected components of binary images, and to derive information about the visibility-problem. that is stated as follows: *for*

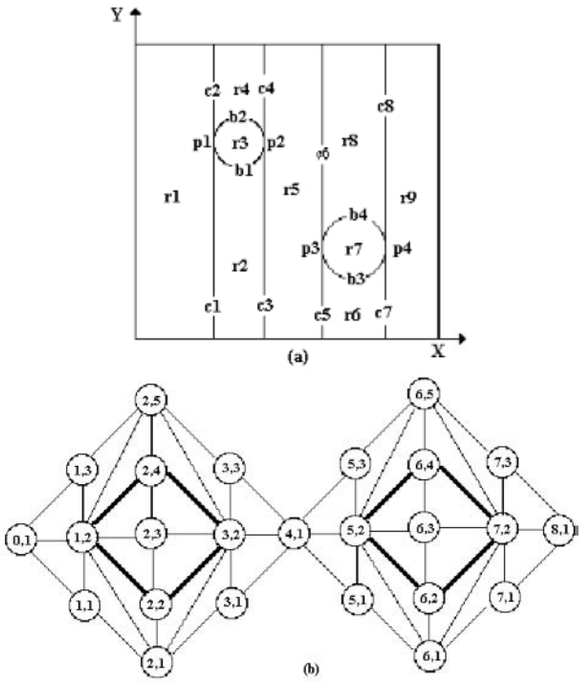

Fig. 2. a) Example of CAD in E^2; b) corresponding Connectivity Graph of the CAD

a given illumination of a scene, with more then one component, decide which component is totally or partially hidden.

The CG derived is planar graph three colorable; its faces are triangles, nodes of which, represent points, bend-points, lines, and $2D$-regions. As said before, each node has three labels $(\lambda, L_\lambda, R_\lambda)$, with $\lambda \in \{p, s, b, c, r\}$, that represent, in the order, its type, left, and right labels.

Proposition 1. *A simple cycle, C, of CG, such that nodes of type p (s) ($L_{p(s)}$ is odd and $R_{p(s)}$ is even), are connected to nodes of type b (L_b and R_b are both even), represents an internal border of a connected component.*

The internal regions of a component of D are obtained as follows:

Proposition 2. *The* internal *region of a component, represented by the cycle C, is the union of the regions, such that their labeling $(\lambda, L_\lambda, R_\lambda)$, $\lambda \in \{p, s\}$, satisfies the condition: L is even and $Rmod4 = 3$ and are connected to some nodes of type p (s) of C. An analogous statement can be formulated for the external regions of a cycle (component), by replacing the condition $Rmod4 = 3$ with $Rmod4 = 1$.*

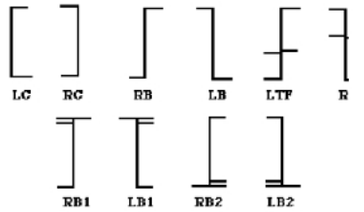

Fig. 3. Basic configurations: left concavity (LC); right concavity (RC); right bend (RB) and left bend (LB); left double point (LTF); right double point (RTF); right bend of type 1 ($RB1$); left bend of type 1 ($LB1$); right bend of type 2 ($RB2$); left bend of type 2 ($LB2$)

Proposition 1 and 2 are straightforward; they follow from the ordering that is settled in the nodes of CG, during the computation of the right and left labels.

In the case of simply connected components the number of cycles is equal to the number of connected components in D. For example, the cycle (bold arcs) in Fig.2b represent the connected components in Fig.2a. Connected components may have N-holes ($N \geq 0$) and the number of components and their holes are retrieved by searching cycles in CG. In this case it is necessary to determine the relation of *inclusion* between cycles. The algorithm generate a graph is a forest of rooted oriented trees, nodes of which represents borders. Oriented arcs represent the relation *to be directly included*. Nodes of the forest represent a component *iff* it belongs to an *even* level. A node of an even level, with N-sons, represents a component with N-holes.

The inclusion relation is easily determined by comparing the highest, and lowest left/right labels of their nodes.

Proposition 3. *Given two simple cycles, C_i, C_j , of CG, representing internal borders of connected components in D, then $C_i \subset C_j$ iff the inequalities:*

$$Lm(C_i) > Lm(C_j) \quad LM(C_i) < LM(C_j)$$

and

$$Rm(C_i) > Rm(C_j) \quad RM(C_i) < RM(C_j)$$

are all satisfied. Where $Lm(C_i)$, $Lm(C_j)$, $LM(C_i)$, $LM(C_j)$, $Rm(C_i)$, $Rm(C_j)$, $RM(C_i)$, $RM(C_j)$ are the lowest and highest left/right labels of the nodes in C_i, and C_j.

Here, we omitted the labeling information to light the notation. This proposition is easily proved by considering again the ordering of the nodes in CG, and by the fact that two cycles, representing regular components in D, cannot have common nodes.

3 Two Views Combination

In the following we will consider the sub-graph of the CG, named *Object CG* (OCG), that is obtained deleting from the CG all nodes and the corresponding arcs representing components of the background of the digital scene (the value 0 is assigned to the background by convention). The label of each node is preserved therefore all results, obtained in the previous section, still hold. The connected components of the OCG represent objects in the scene. The OCG allows us to store less space memory and makes easier the visualization of the results. Moreover, the λ-label will not be explicitly shown in OCG.

In the following we denote by $DCAD_{Y(X)}$ the $DCAD$ along the $Y(X)$-axis and by $OCG_{Y(X)}$ the corresponding graphs. The $DCAD_X$ is performed by scanning the image in a top down and left-right order (see Fig.4).

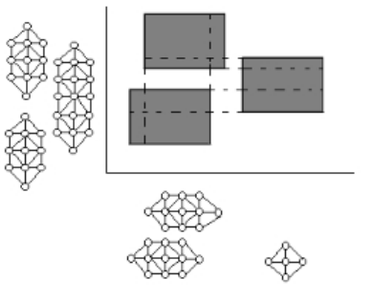

Fig. 4. An example of $DCAD_Y$ and $DCAD_X$ and related graphs

The $DCAD$ of a $2D$ digital scene and the related OCG includes by its definition all topological information regarding the connected components in the scene. However, in the case on non-convex components structural information can be lost or could be ambiguous if we consider only one view.

In the case of binary images most of the shape information is on the borders. The ambiguities, we want to eliminate, regard the positions and orientation of bend points and concavities (these one can be considered the combination of two bend points.

For example, the $DCAD_Y$ of the left and right images in Fig.5 generate the same OCG_Y (see Fig.6), even if the bottom components have a different structure (the bend-points are in the opposite position). Note that the ordering of the λ-labels remains the same in both configurations.

Fig.7 shows the $DCAD_X$ of the previous images. The OCG_X in Fig.s 8a,b corresponds to the left and right images in Fig.7 respectively; note hat in this case the two OCG_X's are different; because the ordering of the λ-labels is not preserved. In the following we will show how to combine the two OCG's to solve this ambiguity.

Fig. 5. $DCAD_Y$ of the left and right images

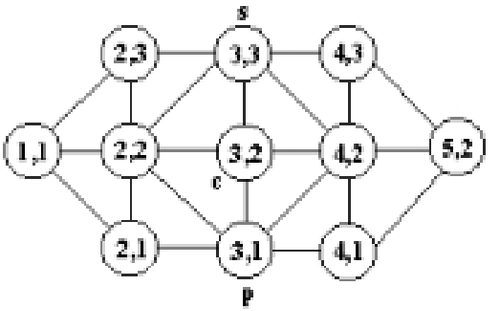

Fig. 6. The OCG_Y of the images in Fig.5

The following properties are used in the algorithm for combining the OCG_Y and the OCG_X graphs.

Proposition 4. *A connected component of an OCG starts with at least one node with label* (p, L, R) *with* L *odd and* R *even, which is connected to nodes* $(\lambda, L+1, R1)$, $(\lambda, L+1, R1+1)$, $(L+1, R1+2)$ *with* $R1 \geq R$ *even.*

This proposition follows from the scanning rules.

Proposition 5. *A sequence of nodes labeled* (p, L, R), $(c, L, R+1)$, $(p, L, R+2)$ *of a connected component of an OCG can be deleted with the arcs and the nodes on the left and right of the triple can be merged as follows:* $\{(\lambda, L-1, x), (\lambda, L+1, x)\} \longrightarrow (\lambda, L-1, x)$ *for* $x \equiv R, R+1, R+2$. *The reduced connected components maintain both topological and structural properties.*

Proof. In fact, the cylinder represented by the nodes (p, L, R), $(c, L, R+1)$, $(p, L, R+2)$ makes a partition of a rectangle into two sub-rectangles that merged do not change neither the topology neither the structure of the image component. The components obtained are said *normalized*.

In the following the sketch of the combining algorithm is given. The time complexity of each step is also evaluated as a function of the linear size n of D and the number of components N_c in the image.

Fig. 7. $DCAD_X$ of the same images

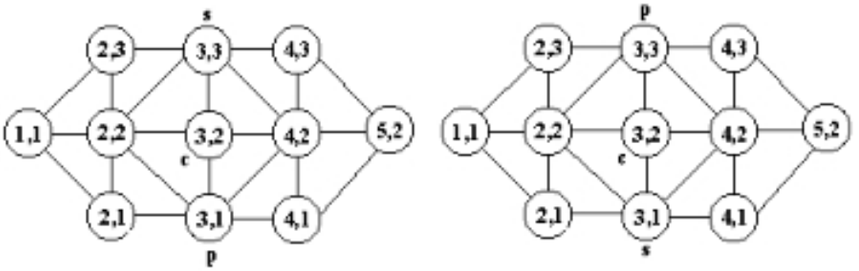

Fig. 8. a) The OCG_X of the left image; b) the OCG_X of the right image

– Algorithm *COMPOUND*

1 *Input* OCG_Y, OCG_X

2 *Components association*
In this step the labeling, generated by the scanning rule, is used. Components in OCG_X are selected by searching for starting nodes that satisfy Proposition 4 that are yet marked and with the lowest labels L and R. The association is made with the corresponding not marked component in OCG_Y that has the highest Y spatial value. The associated components are marked. The association stops after that all components in OCG_X and OCG_Y have been marked. The time complexity of this step is $O(N_c \times N_c)$.

3 *Component normalization*
Using iteratively the reduction rule stated in Proposition 5 performs the normalization of the $OCG_Y(X)$. The time complexity of this step is $O(n \times N_c)$.

4 *Component combination*
The combination of the associated normalized components is performed by considering that for a given odd value of $L > 1$ the sequence of labels is of the form

$$pc(sc)^*p \cup pc(sc)^*s \cup (sc)^*p \cup (sc)^*s$$

Moreover, by using the 9 correspondences between the sequence OCG_Y and OCG_X (see Fig.9 and Table 1) it is possible to set the position of bend-points in each component eliminating the related ambiguities.

5 *Output structural and shape information about each component*
– *end COMPOUND*

The time complexity of this step is $O(n \times N_c)$.

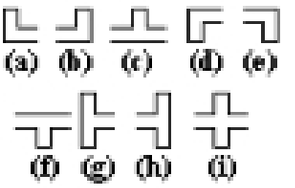

Fig. 9. The 9 correspondences between OCG_Y and OCG_X

Table 1. Words corresponding to configurations in Fig.9

Y	X	$Config.$	Y	X	$Config.$	Y	X	$Config.$
pcs	pcs	$Fig.9a$	pcs	scp	$Fig.9b$	pcs	scs	$Fig.9c$
scp	pcs	$Fig.9d$	scp	scp	$Fig.9e$	scp	scs	$Fig.9f$
scs	pcs	$Fig.9g$	scs	scp	$Fig.9h$	scs	scs	$Fig.9i$

Proposition 6. *The combination phase of the algorithm COMPOUND eliminates ambiguities due to position and orientation of bend points.*

Proof. For binary images, represented in a squared lattice, configurations, given in Fig.9, cover all possible cases. Ambiguities are then solved by parsing left to right the words corresponding to the same object in OCG_X and OCG_Y and testing the occurrencies given in Table 1.

4 Implementation Notes

The serial version of the $DCAD$-algorithm has been implemented in C++ under Linux. The whole computation time of the $COMPOUND$ algorithm is, as we have seen, of the order $O(n \times N_c)$. Note that in most applications $N_c \ll n$. The computation time to perform the $DCAD$ algorithm is of the order $O(n^2)$ and it is the more expensive part of the whole computation. In [12] a parallel version of the $DCAD$-algorithm is given.

The CPU time to run $DCAD$ and $COMPOUND$ algorithms on a PENTIUM III 400MHz was of about $T_{DCAD} = 0.5sec + T_{COMPOUND} = 30msec$ for 256×256 synthetic images containing at most 10 components. The maximum

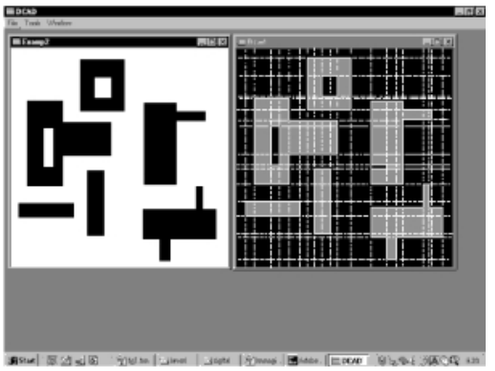

Fig. 10. The $DCAD$ construction of a binary image

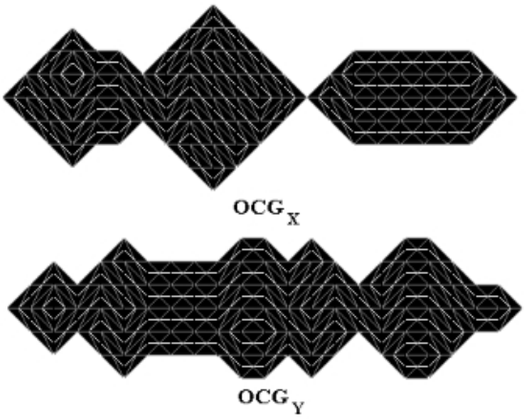

Fig. 11. The OCG_X and OCG_Y of the binary image in Fig.10

number of nodes to be explored was 530 in the case of ten components. The right side of Fig.10 shows the $DCAD$ of a binary image on the left in both views (X, Y). The top and the bottom of Fig.11 show the corresponding OCG_X and OCG_Y respectively.

Either the discrete nature of D or the noise background may generate dummy bend-nodes. They have been removed by using heuristic arguments that are related to the image size; for example discard bend-points with length less than $0.05 \times n$ (e.g. in our case length< 10).

5 Final Remarks

The paper shows a new approach to the analysis of connected components of binary images. The $DCAD$ algorithm allows us to analyse topological properties of binary images. In the paper it is show how to combine the $DCAD$ along two views in order to provide a structural description of digital images. The ordering of the OCG's is at the basis of the search and combining algorithms, allowing them to be tractable and faster. The extension of the $DCAD$ on $3D$ spaces is under study.

References

1. Barcucci, E., Del Lungo, A., Nivat, M., Pinzani, R.: Reconstructing convex polyminoes from horizontal and vertical projections. Theoretical Computer Science **155** (1996) 321–347. 337
2. Herman, G. T.: Image Reconstruction from Projections: The Fundamentals of Computerized Tomography. Academic Press (New York) (1980). 337
3. Chella, A., Di Gesù, V., Infantino, I., Intravaia, D., Valenti, C.: Cooperating Strategy for Objects Recognition. in Lecture Notes in Computer Science book "Shape, contour and grouping in computer vision", Springer Verlag, **1681** (1999) 264–274. 337
4. Collins, G. E.: Quantifier elimination for real closed fields by cylindrical algebraic decomposition. Proc.of the Second GI Conference on Automata Theory and Formal Languages, Springer Lect.Notes Comp. SCi. **33** (1975) 515–532. 337
5. Di Gesù, V. and Renda, R.: An algorithm to analyse connected components of binary images. Geometrical Problems of Image Processing **4** (1991) 87–93. 337, 338
6. Arnon, D. S., McCallum, S.: A polynomial-time algorithm for the topological type of a real algebraic curve. Journal Symb. Comp. **5** (1988) 213–236. 337
7. Arnborg, S., Feng, H.: Algebraic decomposition of regular curves. Journal Symb. Comp. **5** (1988) 131–140. 337
8. Françon, J. M.: Sur la topologie d'un plan arithmétique. Theoretical Computer Science **156** (1996) 31–40. 337
9. Khalimsky, E. D., Kopperman, R., Meyer P. R.: Computer graphics and connected topologies on finite ordered sets. Topology and Applications **36** (1990) 1–17. 337
10. Chassery, J. M. and Montanvert, A.: Geómetrie discreťe en analyse d'images. Hermeś, Paris (1991). 338
11. Kovalevsky, V. A.: Finite Topology as Applied to Image Analysis. Computer Vision, Graphics, and Image Processing **45** (1989) 141–161. 338
12. Chiavetta, F., Di Gesù, V., and Renda, R.: A Parallel Algorithm to analyse connected components on binary images. International Journal of Pattern Recognition and Artificial Intelligence **6**(2,3) (1992) 315–333. 345