Computing the Number of Bubbles and Tunnels of a 3-D Binary Object

Humberto Sossa[1(✉)] and Hermilo Sánchez[2]

[1] Instituto Politécnico Nacional-CIC, Av. Juan de Dios Bátiz S/N, Gustavo a Madero, 07738 Mexico City, Mexico
hsossa@cic.ipn.mx
[2] Centro de Ciencias Básicas, Universidad Autónoma de Aguascalientes, Aguascalientes, Mexico
hsanchez@correo.uaa.mx

Abstract. We present two formulations and two procedures that can be used for computing the number of bubbles and tunnels of a 3-D binary object. The first formulation is useful to determine the number of bubbles of an object, while the second one can be used to calculate the number of tunnels of an object. Both formulations are formally demonstrated. Examples are provided to numerically validate the functioning of both formulations. On the other hand, the first procedure allows obtaining the number of bubbles and tunnels of a 3-D object while the second procedure allows computing the number of bubbles and tunnels of several 3-D objects. Examples with 3-D images are provided to illustrate the utility and validity of the second procedure.

1 Introduction

Many fabricated or natural objects might have bubbles (voids or cavities) and/or tunnels (holes); washers, nuts, some varieties of French or Swiss cheese, sponges, bread, some kind of stones, bones, are some the examples one can mention.

In many image analysis applications, computing the number of bubbles and tunnels for a 3-D object could be an important. It could help in the: (1) analysis of 3-D microstructures of human trabecular bones in relation to its mechanical properties, as is performed in [1], (2) determination of the quantitative morphology and network representation of soil pore structure [2], (3) unambiguous classification of complex microstructures by their three-dimensional parameters applied to graphite in cast iron [3], and (4) analyse the connectivity of the trabecular bone in identifying the deterioration of the bone structure [4].

In this chapter we first introduce two mathematical expressions that allow computing, separately, the number of bubbles (voids) and tunnels (holes) of a 3-D object. These two propositions are also formally demonstrated. In each case, examples are added to numerically validate each proposition. Second, we describe two general methods. The first method allows determining, in two steps, the number of bubbles and tunnels of a 3-D object with both cavities and holes. The second method, permits, in three steps, to accomplish the same task but for several objects into the same image. This chapter is an extended version of the material presented in [42].

A. Fred et al. (Eds.): ICPRAM 2016, LNCS 10163, pp. 194–211, 2017.
DOI: 10.1007/978-3-319-53375-9_11

The rest of the chapter is organized as follows. In Sect. 2, the problem to be faced is stated. In Sect. 3, several related pioneering and recent related methods to compute the Euler number of a digital 3-D image (object) are described. Next, in Sect. 4, several basic definitions that will facilitate the reading of the rest of the paper will be provided. After that, in Sect. 5, the proposed two expressions will be presented and demonstrated. Simple examples are added to numerically validate the operation of each equation. Section 6 will be focused to present and explain the functioning of two general methods for determining the number of bubbles and tunnels of 3-D objects. Section 7 will be devoted to present several examples to show the functioning and applicability of the proposals. In short, Sect. 8 will be oriented to show present the conclusions and directions for further research concerning this investigation.

2 Problem Statement

The problem to be solved in the content of this investigation could be stated as follows: Given a 3-D object composed of face-connected voxels, determine its number of bubbles and tunnels. One way to provide a solution to this problem could be by first computing the Euler number of the 3-D object.

In 3-D, as it known, the Euler number relates the number of bubbles and tunnels of the object. One expression commonly used for this is the following [5, 6]:

$$e = 1 - b_1 + b_2 \tag{1}$$

In this case, b_1 is the number of tunnels or holes of the object and b_2 is its number of bubbles, cavities or voids [7, 8].

It is not difficult to see that Eq. (1) is the simplification of the more general formulation:

$$e = b_0 - b_1 + b_2 \tag{2}$$

where b_0 represents the number of objects in the 3-D binary image $I(x, y, z)$ (for short I) under study. If in the 3-D image there is only one object, then $b_0 = 1$. Just to remember, b_0, b_1 and b_2 are the first three Betti numbers used to distinguish topological spaces based on the connectivity of n-dimensional *simplicial* complexes [9].

The careful reader can rapidly see that Eq. (1) exhibits two problems:

1. The two Betti numbers b_1 and b_2 cannot be obtained by computing local features of the 3-D object. They cannot be got through computing the local features such as the number of vertices or edges. In other words, the computation of Eq. (1) cannot be broken into subtasks, meaning that Eq. (1) cannot be used to compute local measures.
2. Both numbers b_1 and b_2 are part of the same equation. Thus, these two numbers cannot be computed directly from Eq. (1). Indeed, if a 3-D object has both bubbles and tunnels, number b_2 will add up a 1 to Eq. (1) for each bubble found; in the other hand, number b_1 will subtract a 1 to Eq. (1) for each tunnel found. Thus, an 3-D object with exactly the same number of bubbles and the same number of tunnels

will not alter the Euler number of the object because Eq. (1) will produce a 1, due to $(-b_1 + b_2)$ will cancel each other. In Sect. 5, we will explore how to solve these two problems.

3 Related Work

Different methods to compute the Euler number of a 3-D digital object (image) have been reported in literature. In this section, we will briefly describe some of the most important of these works that allow computing this number. We will first describe some of the oldest methods, next we will do the same for some of the most representative recent methods, reported in literature.

3.1 Pioneers Works

One of the first methods introduced to the world to determine the Euler number of a 3-D object was the work reported in [10, 11], but it was only applicable for 6-connectivity. In the first paper, the authors first define what they name "differentials", based on these differentials, the authors present a set of processing algorithms. The algorithms described in [11] can be used for labelling, counting, and computing connected objects in binary three dimensional arrays.

In [12], the authors study the 3-D surface Euler number of a polyhedron based on the Gauss-Bonnet theorem of differential geometry.

Another influential work can be found in [5]. In this work, the authors report several methods to compute the Euler number of a discrete digital image in both 2-D and 3-D.

In [13], the authors introduce an approach to computing the Euler characteristic of a three dimensional digital image by computing the change in numbers of black components, tunnels and cavities in a $3 \times 3 \times 3$ neighbourhood of an object (black) point due to its deletion. In the paper, the authors describe how a parallel implementation of the method is possible using the concept of sub-field [14].

In short, in [15], the authors describe an algorithm for computing the Euler number of a 3D digital image using the topological parameters computed by so called algorithm *topo–para*. This algorithm allows the change in the numbers of object components, tunnels and cavities in the $3 \times 3 \times 3$ neighborhood of a transformed object point (non-object point). Non-object points are obtained by means of a particular transformation acting on the original object points.

Other interesting works concerning the calculation of the Euler number of a 3-D image can be found in [16, 17].

3.2 Recent Works

To begin with this section, in [18, 19], the authors make use of the first two Betti numbers b_0 and b_1 of Eq. (1) for analysing the shape of a 3-D object by first labelling

the points of its skeleton into four types of interest points: boundary, branching, regular and arc points. Authors label skeleton points according to the intersection of its maximal ball with the object. In order to add tolerance to the process, the radius of maximal balls is slightly increased. In a second step, authors make use of reversibility of the skeleton to deduce a labelling of the whole object.

In [20], authors present several fundamental properties of the topological structure of a 3-D digitized picture including the concept of neighbourhood and connectivity among volume cells (voxels) of 3-D digitized binary pictures defined on a cubic grid. They also introduce the concept of simplicial decomposition of a 3-D digitized object. Following this, the authors present two algorithms for calculating the Euler number (genus) of a 3-D figure. The first algorithm is based on the computation of value of a polynomial of binary variables representing voxel densities. The second algorithm utilizes local pattern matching. Both are performed only on the information of a $(2 \times 2 \times 2)$ local space. This method for obtaining the Euler number is an extension of the method for two-dimensional figures given by Gray in [10]. Lobregt et al. [21] derived a similar method for only the 6-connectivity and 26-connectivity cases basing upon a closed netted surface model and the following equation:

$$n - d + f = 2 - q \tag{3}$$

where n is the number of so-called nodal points of the net, d is the number of edges, f is the number of faces of the net and q is the so called connectivity number.

In [22], authors combine integral and digital geometry to develop a method for efficient simultaneous calculation of the intrinsic volumes of sets observed in binary images including surface area, integral of mean curvature, and Euler number. To make this rigorous, the concepts of discretization with respect to an adjacency system and complementarity of adjacency systems are introduced.

In [6], authors describe a method to compute the Euler feature of a 3-D image based on two definitions of foreground run and neighbour number. Following the definitions of foreground run and neighbour number in 2D image [23, 24], they redefine the concepts of foreground run and neighbour number in 3D image. Based on these two concepts, they propose their formula for locally computing the Euler Number of 3D image.

In [25], authors first provide a detailed description of the basics of three-dimensional digital image processing. They then talk about geometric properties of 3D images and 3D image processing fundamentals. After introducing localized processing (filtering), they show how 2D image processing methods are extended to 3D images. Next, they go to the core portion of the research where they first begin with a definition of connectivity and define some fundamental concepts such as topology preservation conditions, Euler numbers, and path and distance functions, thereby leading to some important properties; at the end authors use the ideas developed to present algorithms for processing connected components (for example, labelling, surface/axis thinning, distance transformation, and of course Euler number computation in 3-D.

In short in [26], the authors introduce two equations to compute the Euler of a 3-D object. By using the relationship between contact and enclosing surface concepts, as well as the relationships between vertices, edges and enclosing surfaces, authors derive

an algorithm for obtaining the Euler feature of a 3-D object. It is worth mentioning that this is the most similar work reported in literature to the investigation reported in this chapter.

Other interesting approaches to obtain the Euler number of a 3-D object of 3-D image can be found in [27–30].

4 Definitions

In this section several concepts are defined, these are provided, in the one hand, to help the reader to easily follow the idea behind the proposal. On the other hand, these concept and definition are used to derive and prove the formal theorems baseline of the proposed methods that allow computing the number bubbles and tunnels of a 3-D object.

Definition 1 (voxel). In three dimensions, in the case of a regular grid, a voxel is defined as the cubic unit that makes part of any 3-D object.

Definition 2 (connectivity among voxels). Let v_1 and v_2 to voxels. If v_1 and v_2 share a face, then both voxels are *face connected*; otherwise, if v_1 and v_2 are connected by an edge or by a corner, then they are said to be *edge-connected* or *corner-connected*; respectively, else, v_1 and v_2 they are said to be not connected.

Figure 1 illustrates the four cases provided in Definition 2: Fig. 1(a) shows two voxels connected by a face (*face connected voxels*). In Fig. 1(b) the two voxels are connected by an edge (*edge connected voxels*), while in Fig. 1(c) the same two voxels appear connected by a corner (*corner connected voxels*). In short, Fig. 1(d) depicts the two voxels but disconnected. From now on, in this chapter we will work only with objects connected by faces. We thus have the following definition:

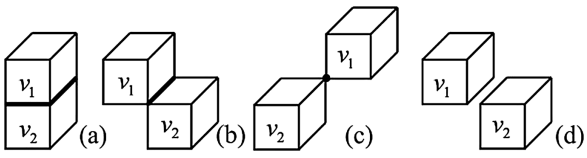

Fig. 1. (a) Two voxels connected by a face. (b) Two voxels connected by an edge. (c) Two voxels connected by a corner. (d) Two not connected voxels.

Definition 3. Any 3-D object O_n composed of n face-connected voxels is any connected region of voxels where all its voxels are only connected by faces.

Figures 2(a) and (b) show two objects connected by their faces. The first object is composed of five voxels, while the second object is composed of seven voxels, respectively.

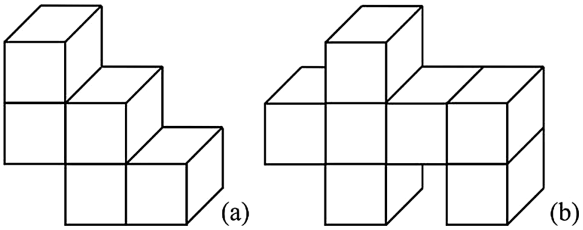

Fig. 2. (a) Object composed of five face-connected voxels; (b) Object composed of seven face-connected voxels.

Definition 4. Let O_n a face-connected 3-D. The faces common to the n pixels of O_n, this the faces interconnecting the n pixels of O_n will be called *contact faces*.

For example, the object shown in Fig. 2(a) has four contact faces, while the object illustrated in Fig. 2(b) has six contact faces.

Definition 5. A *tetra-voxel* is an arrangement of four object voxels as illustrated in Figs. 3(a), (b) or (c). Let *nt* be the number of tetra-voxels that can be found in a 3-D binary image by a simple scanning image method.

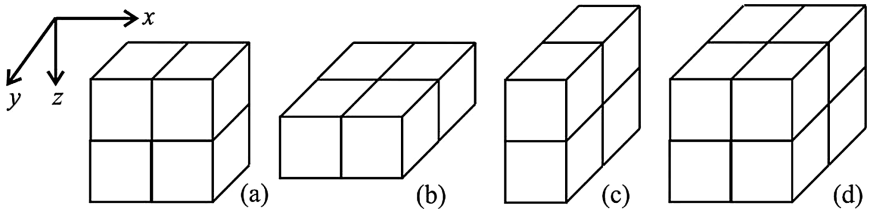

Fig. 3. A tetra-voxel in (a) x direction, (b) y direction, and (c) z direction. (d) An octo-voxel.

Definition 6. An *octo-voxel* is an arrangement of eight object voxels as shown in Fig. 3(d). Let *no* be the number of octo-voxels that can be found in a 3-D binary image by a simple scanning image method.

Due to these two definitions are very important in what follows, let us consider the following four objects given in Fig. 4, composed of 7, 10, 9 and 12 voxels, respectively. If we look at the first two objects, we see that they do not any tetra-voxel or octo-voxel. Now, if we observe at the third object, we note that it contains two tetra-voxels, both in the x direction. Finally, the fourth object, has one octo-voxel and seven tetra-voxels.

Definition 7 (Bubble or cavity). Any connected component of 0-voxels that is not connected to the frame of an image is called a *bubble or cavity*.

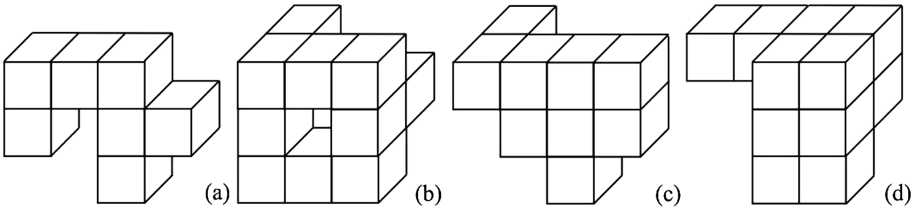

Fig. 4. (a) and (b) Two objects with no tetra-voxels or octo-voxels; (c) An object with two tetra-voxels; (d) An object with one octo-voxel and seven tetra-voxels.

Definition 8 (Simply connected, multiply connected). A connected component of 1-voxels with any holes and cavities is said to be *simply connected*, otherwise, it is said to be *multi connected*.

Definition 9 (Tunnel or hole). Let O_n any multi connected object. A tunnel or a hole is any connected component of 0-voxels that passes through O_n that diminishes the Euler number e of O_n by 1, according to Eq. (1).

To end up with this section, let us consider the following definition of a Connected Component Labelling Algorithm.

Definition 10. A Connected Component Labelling Algorithm (CCLA) applied over an image I assigns a label l to each connected component found in the image I, according to the metric used.

For complete details about Connected Component Labelling Algorithms, the reader is refereed to [31–41].

5 The Proposal: Theoretical Part

Let O_n a 3-D object composed of n face connected voxels for which we want to determine its number of bubbles and it number of tunnels. Let nc, nt, and no, the number of the contact faces, number of tetra-voxels, and number of octo-voxels of O_n, respectively.

5.1 Number of Bubbles of a 3-D Object

Suppose we want to compute the number of bubbles of an object O_n with no tunnels. For this we propose to use the following:

Theorem 1. Let O_n be any connected 3-D binary object composed of n face-connected voxels, the number of bubbles (voids) nb of O_n is given as:

$$nb = (n - nc + nt - no) - 1. \tag{4}$$

Proof. By mathematical induction on the number of voxels of O_n, for the base case: O_1 consisting of a single voxel, $nc = nt = no = 0$, values satisfying Eq. (4).

Induction step: let us assume that Eq. (4) holds for O_n. Let nc', nt', and no' be the number of contact faces, number of tetra-voxels and number of octo-voxels, respectively, of object O_{n+1} that is obtained by adding one voxel to O_n. Let NC, NT and NO be the corresponding numbers for this new voxel. We have that:

$$nc' = nc + NC \tag{5}$$

$$nt' = nt + NT \tag{6}$$

$$no' = no + NO \tag{7}$$

We have to show that Eq. (4) holds for O_{n+1}, i.e.

$$nb' = (n + 1 - nc' + nt' - no') - 1. \tag{8}$$

But this equation can be rewritten as follows:

$$\begin{aligned} nb' &= (n + 1 - nc - NC + nt + NT - no - NO) - 1 \\ &= (n - nc + nt - no) - 1 - NC + NT - NO + 1. \end{aligned} \tag{9}$$

This equation simplifies to:

$$nb' = nb - NC + NT - NO + 1. \tag{10}$$

which we know is true. ∎

To numerically validate Eq. (10), let us consider the 3-D object composed of 25 voxels as shown in Fig. 5(a), with no bubbles and with the central voxel and the central voxel from the upper face missing. For this object, $nb = (26 - 45 + 20) - 1 = 0$ bubbles. Now, suppose that a new voxel is appended to this object as shown in Fig. 5 (b) in such a way that a bubble is obtained. In this case we have that $NC = 4$, $NT = 4$, $NO = 0$ and

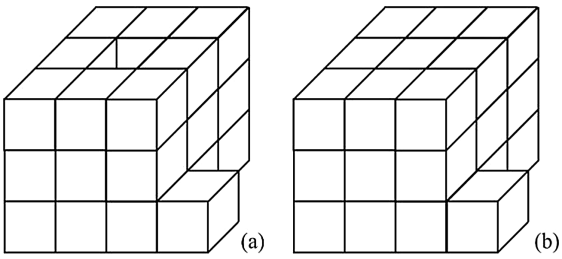

Fig. 5. (a) Object composed of 26 voxels with no bubbles. (b) Object with one bubble after appending a voxel to the object shown in Fig. 5(a).

$$nb' = (n + 1 - nc' + nt' - no') - 1 = (27 - 49 + 24 - 0) - 1 = 1,$$

Also

$$nb' = nb - NC + NT - NO + 1 = 0 - 4 + 4 - 0 + 1 = 1.$$

5.2 Number of Tunnels of a 3-D Object

Suppose now we want to compute the number of tunnels of an object O_n with no bubbles. For this we propose to use the following:

Theorem 2. Let O_n be any connected 3-D binary object composed of n face-connected voxels, the number of tunnels (holes) nh of O_n is given as:

$$nh = 1 - (n - nc + nt - no). \tag{11}$$

Proof. Let us again proceed with the proof by mathematical induction on the number of voxels of O_n. For the base case: O_1 consisting of a single voxel, therefore, we have $nc = nt = no = 0$, values which satisfy Eq. (11).

For the induction step, let us assume that Eq. (11) holds for O_n. Let nc', nt', and no' be the number of contact faces, number of tetra-voxels and number of octo-voxels, respectively, of object O_{n+1} that is obtained by adding one voxel to O_n. Let NC, NT, and NO be the corresponding numbers of this new voxel. We have that:

$$nc' = nc + NC \tag{12}$$

$$nt' = nt + NT \tag{13}$$

$$no' = no + NO \tag{14}$$

It must be shown that Eq. (11) holds for O_{n+1}, i.e.

$$nb' = 1 - (n + 1 - nc' + nt' - no'). \tag{15}$$

But this equation can be rewritten as follows:

$$\begin{aligned} nb' &= 1 - (n + 1 - nc - NC + nt + NT - no - NO) \\ &= 1 - (n - nc + nt - no) + NC - NT + NO - 1. \end{aligned} \tag{16}$$

This equation simplifies to:

$$nb' = nb + NC - NT + NO - 1. \tag{17}$$

which again we know is true. ∎

To numerically validate this last equation, let us consider the 3-D object composed of 15 voxels as shown in Fig. 6(a), with no tunnels. For this object, $nh = 1 - (15 - 14 + 0) = 0$ tunnels. Now, suppose that one new voxel is appended to this object as shown in Fig. 6(b) in such a way that a tunnel is obtained. In this case we have that $NC = 2$, $NT = 0$, $NO = 0$, and

$$nb' = 1 - (n + 1 - nc' + nt' - no') = 1 - (16 - 16 + 0 - 0) = 1$$

Also

$$nb' = nb + NC - NT + NO - 1 = 0 - 2 + 0 - 0 - 1 = 1$$

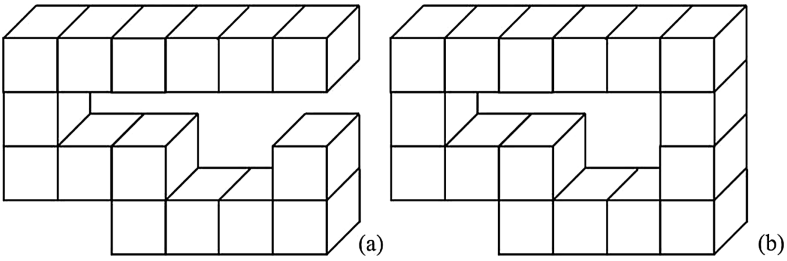

Fig. 6. (a) Object composed of 15 voxels with no tunnels. (b) Object with one tunnel after appending a voxel to the object shown in Fig. 6(a).

6 The Proposal: Practical Part

In this section we present two general procedures for determining the number of bubbles and tunnels of 3-D objects. As we will see, the first procedure is only useful for the case of images with only one object. The second procedure can be used with images containing several objects. Both procedures are only useful for the case of face-connected objects.

6.1 Procedure to Compute the Number of Bubbles and Tunnels of a 3-D Object

Suppose we want to determine the number of bubbles and tunnels of an object O_n. To do this we proceed in two steps as follows. During the first step we obtain the number of bubbles of the object. For this, we utilize of a Connected Component Labelling Algorithm (CCLA) (Definition 10). During the second step we obtain the number of tunnels of O_n by applying Eq. (11). More in detail, given a 3-D image $I(x, y, z)$ of object O_n as shown for example in Fig. 7(a):

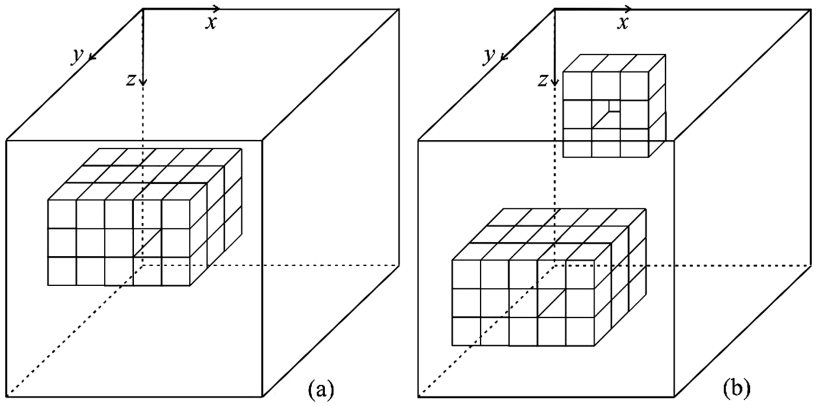

Fig. 7. (a) Image with one objet composed of 41 voxels used to test the functioning of the described procedure in Sect. 6.1 (b) Image with two objects to test the operation of the procedure described in Sect. 6.2.

First Step (Determination of Number of Bubbles):

1. Apply over $I(x, y, z)$ any CCLA over the regions composed of 0-voxels. An adapted version of the algorithm reported in (Gonzalez, 2002), for the 3-D case, can be used for this. Due to bubbles are composed of 0-voxels (Definition 7), this algorithm will output value *ncc*. This variable corresponds to the number of bubbles plus 1. This 1 is obtained because the image background is also labelled; an extra label is generated.
2. Compute the number of bubbles of object O_n, *nb*, as $ncc - 1$.

Second Step (Determination of the Number of Tunnels):

1. Apply Eq. (11) over image $I(x, y, z)$. If the object has bubbles and tunnels, this application will produce the number of tunnels minus the number of bubbles nh_b. Refer to Eq. (1).
2. Add to the result obtained in the last step to get the number of tunnels $nh = nh\_b + nb$ of object O_n.

To numerically validate the above described procedure, let us consider the image with one object as shown in Fig. 7(a), composed of 41 voxels, one bubble and one tunnel. The bubble is the 0-voxel in the centre of the second vertical slice of 1-voxels of the object (left to right). The tunnel is composed of the three 0-voxels along the fourth vertical slice of 1-voxels of the object (left to right).

The first step of the above described algorithm, outputs: $nb = 1$, while the second step outputs $nh = nh\_b + nb = 1 - (41 - 76 + 36 + 0) + 1 = 0 + 1$, as desired. Note that $nh\_b = 0$ due to the object has one bubble and one tunnel, that according to Eq. (1) they cancel each other.

6.2 Procedure to Compute the Number of Bubbles and Tunnels of a Set of 3-D Objects

Suppose now are given an image $I(x, y, z)$ of b_0 voxelized objects; for each of these b_0 objects we would like to compute their numbers of bubbles and tunnels, respectively.

In this case we would need to apply a similar procedure as described in Sect. 6.1 with an additional step. We proceed into three steps as follows:

1. Apply any CCLA over image $I(x, y, z)$ to obtain b_0 labelled connected 3-D objects.
2. Apply the first step of the procedure described in Sect. 6.1 to each labelled connected region $R_i, i = 1, 2, \ldots, b_0$. For each R_i we obtain its number of bubbles $nb_i, i = 1, 2 \ldots, b_0$.
3. Apply the second step of the same procedure described in Sect. 6.1 to obtain the number of tunnels of each object.

To numerically validate the above described procedure, let us consider Fig. 7(b) with two objects as depicted. As can be seen, the first object is composed of 8 voxels and one tunnel and the second one integrated of 41 voxels, with one bubble and one tunnel (the same object of Fig. 7(a)).

The first step of the above described procedure provides as a result two labels (two connected 3-D regions).

Next, for each labelled (region), the second step obtains $nb_1 = 0$ and $nb_2 = 1$, respectively. Finally, the third step outputs $nh_1 = nh\_b_1 + nb_1 = 1 - (8 - 8 + 0 - 0) + 0 = 1 + 0 = 1$ for the first object and $nh_2 = nh\_b_2 + nb_2 = 1 - (41 - 76 + 36 + 0) + 1 = 0 + 1 = 1$, for the second object, as desired.

7 Results and Discussion

In this section we report four experiments to validate the performance of our proposal. We only validate the correct functioning of the general algorithm described in Sect. 6.2.

7.1 First Experiment

During the first experiment, we utilized five 3-D images of size $100 \times 100 \times 100$ voxels. Each image was designed to have a different number of objects as established in row two of Table 1. Each time an object was added to the image it was added manually to have a control over the number of its number of bubbles and holes. Rows 3 and 4 of Table 1 show the correct number of bubbles and tunnels (Correct *nb* and Correct *nh*) of each object of each image, respectively.

The procedure described in Sect. 6.2 was applied to each of the five images. It was programed in Java NetBeans with the Processing Applet in a desktop computer with a Core i7 model 2600 processor with 8 Gb of RAM. Rows 5 and 6 depict the computed number of bubbles and tunnels for object of each image, respectively. From these rows note also that in all cases, as expected, the correct values, *nb* and *nh*, for each object were produced by the procedure. The average time to compute the number of bubbles

Table 1. Results obtained by the application of the procedure described in Sect. 6.2 to the five selected images.

Image number	1	2	3	4	5
Number of objects per image	1	2	3	4	5
Correct *nb*	2	2,1	2,1,3	2,1,3,2	2,1,3,2,5
Correct *nh*	1	1,3	1,3,1	1,3,1,2	1,3,1,2,4
Computed *nb*	2	2,1	2,1,3	2,1,3,2	2,1,3,2,5
Computed *nh*	1	1,3	1,3,1	1,3,1,2	1,3,1,2,4

and tunnels of each of the b_0 objects in image I was 29.6 ms. It is worth mentioning that most of time is consumed by the connected component algorithms.

7.2 Second Experiment

During this experiment, we studied if the number of object-voxels influenced computation time when the total procedure was applied over an image. For this, we automatically generated a set of images with an increasing number of object-voxels. We defined a variable (nv) telling how many object-voxels will appear in the image. When $nv = 0.0$, it meant that the corresponding image will have only background voxels, for $nv = 0.05$, it meant that 5% of the generated voxels will belong to objects, and so on. Each time we increased variable nv by 0.05. For each value of variable nv we generated 10 images. We took the average time to fully process the whole set of 210 images and computed the average time. With the exception of the first case, in average the time consumed by the connected component algorithm was of 25.5 ms.

7.3 Third Experiment

In this case, we demonstrated the applicability of our method when applied to objects of various shapes and complexities. Figure 8 shows six of these objects: (a) a sphere, (b) a vase, (c) a torus, (d) a cheese, a (e) dragon and (f) a bookcase. Second and third row of Table 2 show the true values of number of bubbles and tunnels of each the six objects, while fourth and fifth rows show the computed values. As expected it can be seen that in all six cases, the computed values coincide with the true values. The average time to obtain the desired results was of 55.5 ms.

7.4 Fourth Experiment

In this last experiment we demonstrated the robustness of our method to image transformations such as translations, rotations and scale changes. For this, we took one of the objects (in this case one of the objects with a cheese shape) and translated, rotated and scaled inside its image. Four of these transformed versions are depicted in Fig. 9. Again, in all cases, one can easily verify that if the object remains face connected after being transformed, the desired number of tunnels was correctly computed.

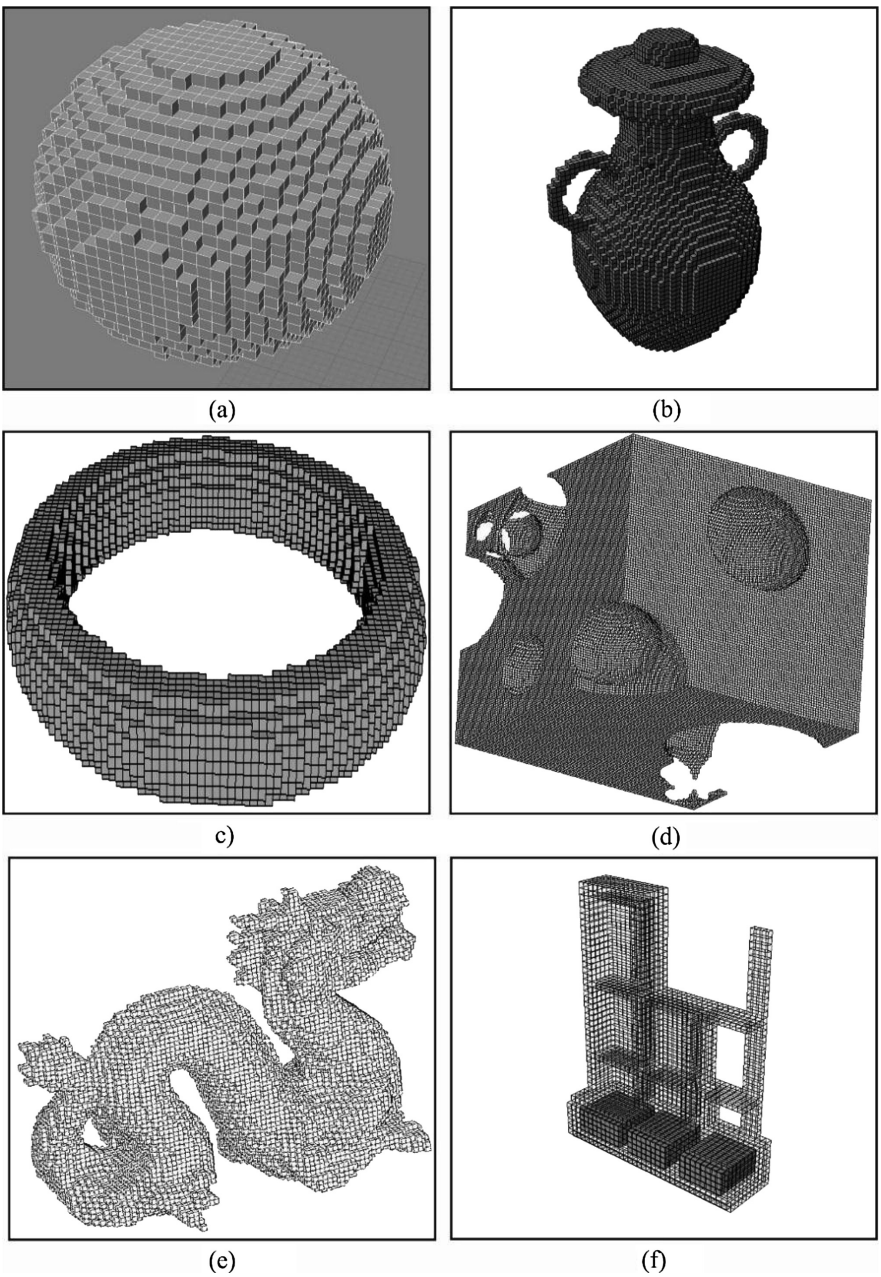

Fig. 8. Six objects of different 3-D objects with different complexity to demonstrate the applicability of the proposal. (a) a sphere, (b) a vase, (c) a torus, (d) a cheese, a (e) dragon and (f) a bookcase.

Table 2. Results obtained by the application of the procedure to the six objects of Fig. 8.

Object	Sphere	Base	Torus	Cheese	Dragon	Bookcase
Correct number of bubbles *nb*	0	0	0	1	0	3
Correct number of tunnels *nh*	0	2	1	4	1	2
Computed number of bubbles *nb*	0	0	0	1	0	3
Computed number of tunnels *nh*	0	2	1	4	1	2

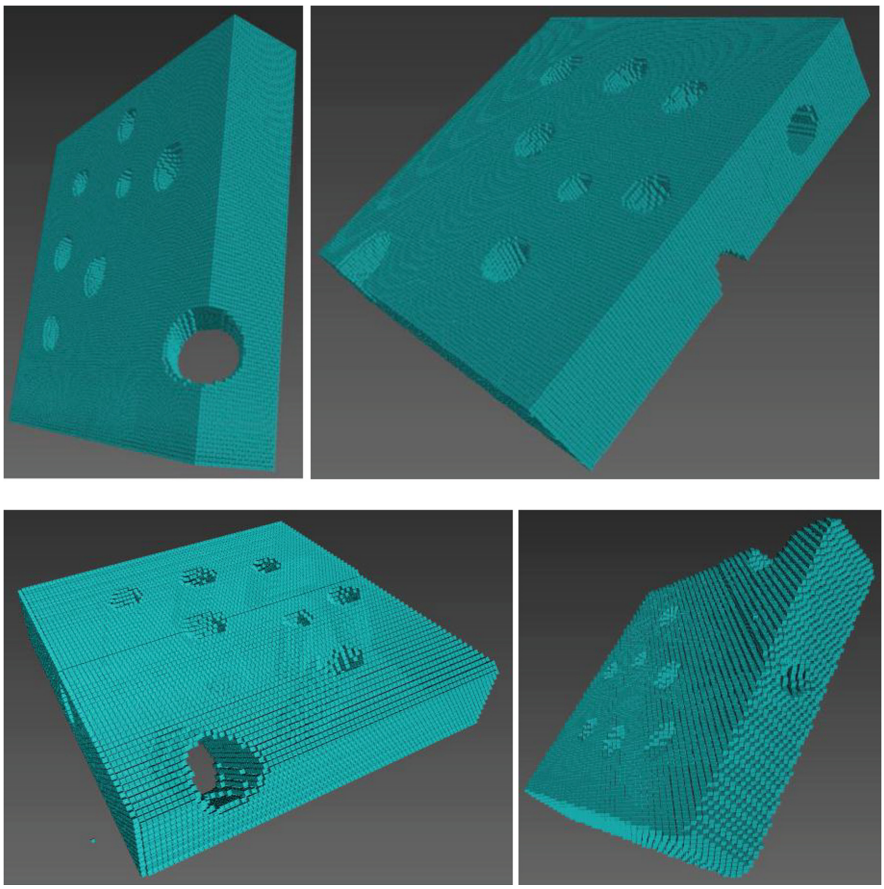

Fig. 9. Four transformed versions of the cheese object.

8 Conclusions and Directions for Further Research

In this paper we have presented two theoretical formulations and two general procedures to obtain the number of voids and tunnels of 3-D objects. The first formulation ((Eq. (4)) allows computing the number of bubbles of a 3-D binary face connected

object; the second formulation (Eq. (11))) is useful for determining the number of tunnels. Both formulations have been formally demonstrated. In both cases, numerical examples were provided to numerically validate both equations.

Based on the above two formulations, two general procedures have been proposed. The first procedure, fully described in detail in Sect. 6.1, permits calculating the number of bubbles and tunnels of a 3-D binary face connected object from its binary image. The second general procedure, completely explained in Sect. 6.2, allows to produce the same but for any number of voxelixed objects. Experimental results with images of objects of different sizes and complexities have been given to show the applicability of both procedures.

Through the experiments, we have observed that the time spent in milliseconds expended by our method is reduced making it to be used in real time applications.

Considering that in some real applications, images of larger dimensions can be found, we propose for further work to implement our proposed procedure described in Sect. 6.2 into a FPGA or a GPU processor to determine how much processing times can be reduced.

Acknowledgements. H. Sossa thanks COFAA-IPN, SIP-IPN and CONACYT under Grants 20151187, 20161126, 155014 and 65 (Frontiers of Science), respectively, for the economic support to carry out this research. E. Sánchez thanks the Centro de Ciencias Básicas of the Universidad Autónoma de Aguascalientes for the support.

References

1. Uchiyama, T., Tanizawa, T., Muramatsu, H., Endo, N., Takahashi, H.E., Hara, T.: Three-dimensional microstructural analysis of human trabecular bone in relation to its mechanical properties. Bone **25**(4), 487–491 (1989)
2. Vogel, H.J., Roth, K.: Quantitative morphology and network representation of soil pore structure. Adv. Water Resour. **24**, 233–242 (2001)
3. Velichko, A., Holzapfel, C., Siefers, A., Schladitz, K., Mücklich, F.: Unambiguous classification of complex microstructures by their three-dimensional parameters applied to graphite in cast iron. Acta Mater. **56**, 1981–1990 (2008)
4. Roque, W.L., de Souza, A.C.A., Barbieri, D.X.: The Euler-Poincaré characteristic applied to identify low bone density from vertebral tomographic images. Rev. Bras. Reumatol. **49**(2), 140–152 (2009)
5. Kong, T.Y., Rosenfeld, A.: Digital topology: introduction and survey. Comput. Vis. Graph. Image Process. **48**, 357–393 (1989)
6. Lin, X., Xiang, Sh., Gu, Y.: A new approach to compute the Euler number of 3D image. In: 3rd IEEE Conference on Industrial Electronics and Applications, pp. 1543–1546 (2008)
7. Lee, C.N., et al.: Winding and Euler numbers for 2D and 3D digital images, CVGIP: graph. Models Image Process. **53**(6), 522–537 (1991)
8. Lee, C.N.: Holes and genus of 2D and 3D digital images, CVGIP: graph. Models Image Process. **55**(1), 20–47 (1993)
9. Delfinado, C.J.A., Edelsbrunner, H.: An incremental algorithm for Betti numbers of simplicial complexes on the 3-sphere. Comput. Aided Geom. Des. **12**(7), 771–784 (1995)

10. Gray, S.B.: Local properties of binary images in two and three dimensions. IEEE Trans. Comput. **C-20**(5), 551–561 (1971)
11. Park, C.M., Rosenfeld, A.: Connectivity and genus in three dimension, TR-156. Computer Vision Laboratory, Computer Science Center, University of Maryland, College Park, MD (1971)
12. Lee, C.N., Rosenfeld, A.: Computing the Euler number of a 3d image. In: Proceedings of the IEEE First International Conference on Computer Vision, pp. 567–571 (1987)
13. Saha, P.K., Chaudhuri, B.B.: A new approach to computing the Euler characteristic. Pattern Recogn. **28**(12), 1955–1963 (1995)
14. Golay, M.J.E.: Hexagonal parallel pattern transformations. IEEE Trans. Comput. **C-18**, 733–740 (1969)
15. Saha, P.K., Chaudhuri, B.B.: 3D Digital topology under binary transformation with applications. Comput. Vis. Image Underst. **63**(3), 418–429 (1996)
16. Morgenthaler, D.G.: Three-dimensional digital topology: the genus. TR-980 November 1980. Computer Vision Laboratory Computer Science Center B University of Maryland (1980)
17. Morgenthaler, D.G., Rosenfeld, A.: Surfaces in three-dimensional digital images. Inf. Control **51**, 227–247 (1981)
18. Bonnassie, A., et al.: Shape description of three-dimensional images based on medial axis. In: Proceedings of the 2001 International Conference on Image Processing, pp. 931–934 (2001)
19. Bonnassie, A., Peyrin, F., Attali, D.: A new method for analyzing local shape in three-dimensional images based on medial axis transformation. IEEE Trans. Syst. Man Cybern. Part B Cybern. **33**(4), 700–705 (2003)
20. Toriwaki, J., Yonekura, T.: Euler number and connectivity indexes of a three dimensional digital picture. Forma **17**, 183–209 (2002)
21. Lobregt, S., Verbeek, P.W., Groen, F.C.A.: Three-dimensional skeletonization: principle and algorithm. IEEE Trans. Pattern Anal. Mach. Intell. **PQMI-2**(1), 75–77 (1980)
22. Schladitz, K., Ohser, J., Nagel, W.: Measuring intrinsic volumes in digital 3d images. In: Kuba, A., Nyúl, László, G., Palágyi, K. (eds.) DGCI 2006. LNCS, vol. 4245, pp. 247–258. Springer, Heidelberg (2006). doi:10.1007/11907350_21
23. Lin, X.Z., Sha, Y., Ji, J.W., Wan, J.B.: Image Euler number calculating for intelligent counting. In: Proceedings of the Seventh International Conference on Electronic Measurement and Instruments (ICEMI 2005), August 2005, vol. 8, pp. 642–645. International Academic Publishers/World Publishing Corporation, Beijing (2005)
24. Lin, X.Z., Wang, Y.M., Sha, Y., Ji, J.W.: A new approach to compute the Euler number of 2d image. In: Proceedings of the International Conference on Sensing, Computing and Automation (ICSCA 2006), May 2006, pp. 1376–1379. Watam Press, Chongqing (2006)
25. Toriwaky, J., Yoshida, H.: Fundamentals of Digital Image Processing. Springer, Heidelberg (2009)
26. Sánchez, H., Sossa, H., Braumann, U.-D., Bribiesca, E.: The Euler-Poincaré formula through contact surfaces of voxelized objects. J. Appl. Res. Technol. **11**, 55–78 (2013)
27. Yonekura, T., Toriwaki, J., Fukumura, T., Yokoi, S.: Topological properties of three-dimensional digitized picture data (1)-connectivity and Euler number. Paper of the Professional Group on Pattern Recognition and Learning, The Institute of Electronics and Communication Engineers of Japan (IECE, Denshi-Tsushin Gakkai), PRL80-1 (1980). (in Japanese)
28. Yonekura, T., Toriwaki, J., Fukumura, T., Yokoi, S.: On connectivity and Euler Number of three dimensional digitized binary pictures. Trans. IECE **E63**(11), 815–816 (1980). Japan

29. Yonekura, T., Yokoi, S., Toriwaki, J., Fukumura, T.: Connectivity and Euler number of figures in the digitized three-dimensional space. Trans. IECE (Denshi-Tsushin Gakkai Ronbunshi), Japan, **J65-D**(1), 80–87 (1982). (in Japanese)
30. Ohser, J., Nagel, W., Schladitz, K.: The Euler number of discretised sets – surprising results in three dimensions. Image Anal. Stereol. **22**, 11–19 (2003)
31. Gonzalez, R., Woods, R.: Digital Image Processing. Prentice Hall, Upper Saddle River (2002)
32. He, L., Chao, Y., Suzuki, K.: A survey of labeling algorithms. In: Joint 4th International Conference on Soft Computing and Intelligent Systems and 9th International Symposium on advanced Intelligent Systems (SCIS & ISIS 2008), September 17–21, 2008, Nagoya University, Nagoya, Japan, pp. 1293–1298 (2008)
33. He, L., Chao, Y., Suzuki, K.: An efficient first-scan method for label-equivalence-based labeling algorithms. Pattern Recogn. Lett. **31**, 28–35 (2010)
34. He, L., Chao, Y., Suzuki, K.: A run-based one and a half scan connected-component labeling algorithm. Int. J Pattern Recogn. Artif. Intell. **24**, 557–579 (2010)
35. Sutheebanjard, Ph., Premchaiswadi, W.: Efficient scan mask techniques for connected components labeling algorithm. EURASIP J. Image Video Process. **2011**, 14 (2011). doi:10.1186/1687-5281-2011-14
36. Lifeng, H., Xiao, Z., Suzuki, K.: A new two-scan algorithm for labeling connected components in binary images. In: Proceedings of the World Congress on Engineering, London, UK, 4–6 July 2012, pp. 1141–1146 (2012)
37. Grana, C., Montangero, M., Borghesani, D.: Optimal decision trees for local image processing algorithms. Pattern Recogn. Lett. **33**, 2302–2310 (2012)
38. He, L.F., Chao, Y., Suzuki, K.: An algorithm for connected-component labeling, hole labeling and Euler number computing. J. Comput. Sci. Technol. **28**, 468–478 (2013)
39. Lifeng, H., Xiao, Z., Yuyan, C., Suzuki, K.: Configuration-transition-based connected-component labeling. IEEE Trans. Image Process. **23**, 943–951 (2014)
40. Lifeng, H., Yuyan, C.: A very fast algorithm for simultaneously performing connected-component labeling and Euler number computing. IEEE Trans. Image Process. **24**, 2725–2735 (2015)
41. Chang, W.-Y., Chiu, Ch.-Ch., Yang, J.-H.: Block-based connected-component labeling algorithm using binary decision trees. Sensors **15**, 23763–23787 (2015)
42. Soosa, H.: On the computation of the number of bubbles and tunnels of a 3-d binary object. In: 5th International Conference on Pattern Recognition Applications and Methods, Roma, Italia, 24–26 February, pp. 17–23 (2016)