Algorithm 964: An Efficient Algorithm to Compute the Genus of Discrete Surfaces and Applications to Turbulent Flows

ADRIÁN LOZANO-DURÁN and GUILLEM BORRELL, Universidad Politécnica de Madrid

A simple and efficient algorithm to numerically compute the genus of surfaces of three-dimensional objects using the Euler characteristic formula is presented. The algorithm applies to objects obtained by thresholding a scalar field in a structured-collocated grid and does not require any triangulation of the data. This makes the algorithm fast, memory efficient, and suitable for large datasets. Applications to the characterization of complex surfaces in turbulent flows are presented to illustrate the method.

Categories and Subject Descriptors: G.4 [**Mathematical Software**]: Algorithm Design and Analysis; I.3 [**Computational Geometry and Object Modeling**]; J.2 [**Physical Science and Engineering**]: Physics

General Terms: Algorithms

Additional Key Words and Phrases: Genus, Euler characteristic, voxels, turbulence, coherent structures, turbulent/nonturbulent interface

ACM Reference Format:
Adrián Lozano-Durán and Guillem Borrell. 2016. An efficient algorithm to compute the genus of discrete surfaces and applications to turbulent flows. ACM Trans. Math. Softw. 42, 4, Article 34 (June 2016), 19 pages.
DOI: http://dx.doi.org/10.1145/2845076

1. INTRODUCTION

We present a fast and memory-efficient algorithm to numerically compute the topological genus of all surfaces associated with three-dimensional objects in a discrete space. The article is aimed at the turbulence community interested in the topology of three-dimensional entities in turbulent flows such as coherent structures [del Álamo et al. 2006; Lozano-Durán et al. 2012] or turbulent/nonturbulent interfaces [da Silva et al. 2014a]. Konkle et al. [2003] describes fast methods for computing the genus of triangulated surfaces, which usually is a time- and memory-consuming process. Our algorithm does not rely on triangulation [Toriwaki and Yonekura 2002; Chen and Rong 2010; Ayala et al. 2012; Cruz and Ayala 2013] and is adapted to exploit the structured-collocated grid commonly used in the largest direct numerical simulations of turbulent flows [Kaneda et al. 2003; Hoyas and Jiménez 2008; Sillero et al. 2013]. Our goal is to provide a clear and easy description of the algorithm and sample codes. See Computational Fluid Mechanics Lab [2015] for more examples in Fortran and Python.

This work was supported by the Computational Fluid Mechanics Lab headed by Javier Jiménez, the European Research Council, under grant ERC-2010.AdG-20100224. A. Lozano-Durán was supported by an FPI fellowship from the Spanish Ministry of Education and Science and the ERC.
Authors' addresses: A. Lozano-Durán, Center for Turbulence Research, Stanford University, Stanford, CA 94305, USA; email: adrianld@stanford.edu; G. Borrell, Technical University of Madrid, Plaza Cardenal Cisneros 3, Spain; email: guillem@torroja.dmt.upm.es.

DOI: http://dx.doi.org/10.1145/2845076

The genus is a topologically invariant property of a surface defined as the largest number of nonintersecting simple closed curves that can be drawn on the surface without separating it. The genus is negative when applied to a group of several isolated surfaces, as it is considered that no closed curves are required to separate them. Both spheres and discs have genus zero, whereas a torus has genus one. On the other hand, two separated spheres or the surfaces defined by a sphere shell (or sphere with an internal cavity) has genus minus one. For a set of objects in a given region, the genus is equal to the *number of holes – number of objects – number of internal cavities*+ 1. The concept is also defined for higher dimensions, but the present work is restricted to two-dimensional surfaces embedded in a three-dimensional space. In integral geometry, the genus is part of a larger set of Galilean invariants called *Minkowski functionals,* which characterize the global aspects of a structure in a n-dimensional space. The genus is also closely related to the Betti numbers, and more details can be found in Thompson [1996].

Regarding its applications, the genus has proven to be very useful to characterize a wide variety of structures in many fields, such as cosmology and related cosmic microwave background studies [Einasto et al. 2007]. The large-scale structure of the universe has been studied over the years through analyses of the distribution of galaxies in three dimensions using the genus for characterizing its topology [Gott et al. 1986, 1987, 1989; Hamilton et al. 1986' Vogeley et al. 1994; Mecke et al. 1994; Park et al. 2005a, 2005b]. For a given threshold of the galaxy density, an isosurface separating higher and lower density regions is defined and the genus of such contour evaluated. This allows one to compare the topology observed with that expected for Gaussian random phase initial conditions [Guth 1981; Linde 1983]. In all of these applications, the computation of the genus was performed by calculating the discrete integrated Gaussian curvatures [Gott et al. 1986; Chen and Rong 2010] following the Fortran algorithm by Weinberg [1988] based on the Gauss-Bonnet theorem. As we will show in Section 3, the present method does not rely on computing any curvatures.

Other applications are oriented to medical and biological areas and use the genus of surfaces or three-dimensional objects—for example, to compute adenine properties in the biochemistry field [Konkle et al. 2003] and to evaluate the osteoporosis degree of mice femur [Martin-Badosa et al. 2003] or human vertebrae [Odgaard and Gundersen 1993].

The Minkowski functionals have been introduced in the study of turbulent flows through the so-called shapefinders [Sahni et al. 1998]. Leung et al. [2012] studied the topological properties of enstrophy isosurfaces in isotropic turbulence by filtering the data at different scales and computing structures of high enstrophy together with its corresponding Minkowski functionals. The geometry of the educed objects was then classified with two nondimensional quantities—planarity and filamentarity—that measure the shape of the structures.

Borrell and Jiménez [2013] followed a strategy based on the genus to decide optimal thresholds in turbulent/nonturbulent interfaces extracted from numerical data. Several surfaces were obtained by thresholding the fluctuating enstrophy field in a turbulent boundary layer, and their associated genus was used as an indicator of the complexity of the interface. This topological description was crucial to decide the range of thresholds where a vorticity isocontour can be considered a turbulent/nonturbulent interface.

The rest of the article is organized as follows. Important definitions are provided in Section 2. The algorithm to compute the genus is described in Section 3. An alternative method is presented in Section 4 and validated with the previous one in Section 5, which also contains some scalability tests. Two applications to turbulent flows are shown in Section 6. Conclusions are offered in Section 7.

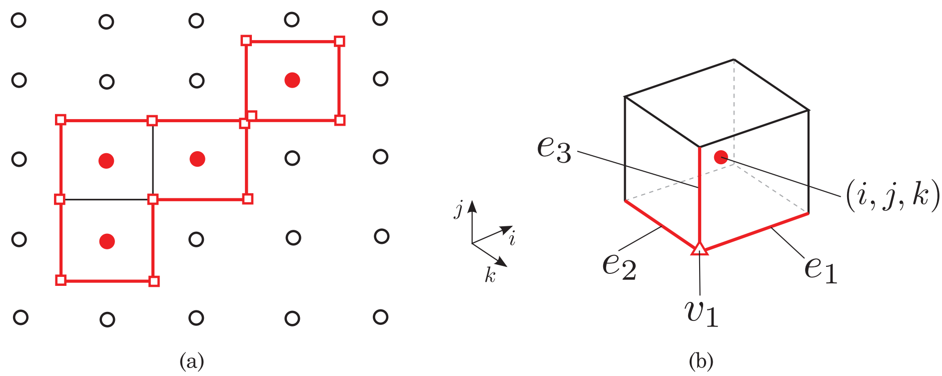


Fig. 1. (a) Two-dimensional example of a structured-collocated grid defined by the open and closed circles. Points satisfying relation (1) are red closed circles, and their corresponding voxels are solid lines. The exterior edges are the thicker lines colored in red, and the exterior vertices are marked by squares. Objects are built by connecting orthogonal neighboring points with $A = 1$, which results in two objects in this particular example. (b) Voxel around the grid point (i, j, k). Only edges e_1, e_2, and e_3 (red lines) and vertex v_1 (triangle) are taken into account to compute the number of exterior vertices and edges corresponding to the voxel centered at (i, j, k).

2. DEFINITIONS

We will first introduce the definitions of *object*, *voxel*, *surface*, *hole*, *cavity*, and *genus*. The starting point is a discrete three-dimensional scalar field, $\phi = \phi(i, j, k)$, with $i = 1, \ldots, n_x$, $j = 1, \ldots, n_y$, and $k = 1, \ldots, n_z$, where n_x, n_y, and n_z are the number of grid points in each direction, respectively, separated by a grid spacing δ. Given a thresholding value α, we define the points belonging to the three-dimensional objects as those satisfying

$$\phi(i, j, k) > \alpha, \tag{1}$$

which can be expressed as a scalar field $A = A(i, j, k)$ whose values are equal to one at (i, j, k) if relation (1) is satisfied and zero otherwise. The latter is referred to as an empty region.

Three-dimensional individual objects in A are constructed by connecting neighboring points with value 1. Figure 1(a) shows a two-dimensional example. Connectivity is defined in terms of the six orthogonal neighbors in the grid, usually called *6-connectivity*. Points contiguous in oblique directions are not directly connected, although they may become so indirectly through connections with other points. This remark is important, as the 6-connectivity is built-in in the algorithm, and, for instance, the number of objects in the example shown in Figure 1(a) is not one but two.

We define the voxel associated with $A(i, j, k) = 1$ as the cube centered at (i, j, k) and with edge length equal to δ (see Figure 1(b)). For a given object, its surface is delimited by the exterior faces of its voxels (i.e., those facing empty regions). In the two-dimensional example shown in Figure 1(a), the one-dimensional "surface" is highlighted with red lines. Actual three-dimensional examples are shown later in Figure 4. A hole is a empty region piercing the object, as the torus in Figure 4(a), and a cavity an internal empty region that is locally not connected to the exterior. The term *handle* will be used occasionally as a synonym for *hole*, as they are topologically equivalent.

Our goal is to compute the genus of all surfaces contained in A. Mathematically, the genus g is defined in terms of the Euler characteristic χ via the relationship

$$\chi = 2 - 2g. \tag{2}$$

The Euler characteristic can be calculated for continuous surfaces as

$$\chi = \frac{1}{2\pi} \iint_{\Sigma} K \mathrm{d}\Sigma, \tag{3}$$

where K is the Gaussian curvature of all objects considered and Σ their area. However, we are more interested in the original discrete form for polyhedral surfaces,

$$\chi = F - E + V, \tag{4}$$

where F, E, and V respectively are the number of exterior faces, edges, and vertices of all polyhedra. In this case, the curvature can be considered to be located at the discrete edges, but the calculations lead to the same results as (3). The connection between the discrete and continuous formulations is the Gauss-Bonnet theorem [Chavel 2006]. Intuitively, in terms of the elements defined earlier, the genus is equal to the *number of holes – number of objects – number of internal cavities* $+1$.

3. ALGORITHM

The present algorithm exploits formula (4) and the structured-collocated nature of the data to compute the genus of all surfaces contained in the three-dimensional space defined by the scalar field A, without previous triangulation or calculation of the Gaussian curvatures. Note that this differs from other works that compute the genus of the three-dimensional objects themselves [Toriwaki and Yonekura 2002; Chen and Rong 2010; Ayala et al. 2012; Cruz and Ayala 2013]. The method is conceived for large datasets of the order of 10^2 GiB and takes A as input.

First, we provide a general description of the algorithm. The key idea is to place a voxel around every (i, j, k) point with $A(i, j, k) = 1$, as the example shown in Figure 1(b), and to create a virtual mesh using the exterior elements of the resulting polyhedra. The term *virtual* is used here in the sense that no actual faces, vertices, or edges have to be stored for each object—that is, there is no actual structure in the code to do so, in contrast to the standard meshes obtained by triangulation, where these are saved in a file or in memory for all objects. Figure 1(a) shows an example of a virtual mesh in a two-dimensional case.

The way to proceed is then to compute the Euler characteristic of the virtual mesh and thereafter the genus. The value of χ is easily calculated once the total number of exterior faces, vertices, and edges are known for all objects within A. To achieve this, three variables F, V, and E are used to store the total number of exterior faces, vertices, and edges, respectively, which are counted looping once through the array A. At each (i, j, k), a voxel is placed if $A(i, j, k) = 1$ and the counters F, V, and E (initially set to zero) are increased accordingly every time faces, edges, and vertices are identified as exterior. The selection of edges and vertices taken into account at each (i, j, k), shown in Figure 1(b), is deliberately chosen to avoid counting several times edges and vertices already considered.

To prevent any problems at the boundaries of the field A, the original grid is extended by padding two extra planes of zeros at the beginning and at the end of each dimension. The new field will still be called A, but now with dimensions $N_x = n_x + 4$, $N_y = n_y + 4$, and $N_z = n_z + 4$. For simplicity, we consider that A is fully loaded in memory, but note that this is not required and it could be loaded in small chunks or planes.

A more detailed description of the algorithm is now presented:

(1) *Initialize the variables*. F, V, and E are integers containing the number of exterior faces, vertices, and edges, and they initially are set to zero. For large cases, they must be double precision. $A(i, j, k)$ is the array whose points are set to one if they belong to an object and to zero otherwise. N_x, N_y, and N_z are the sizes of A after

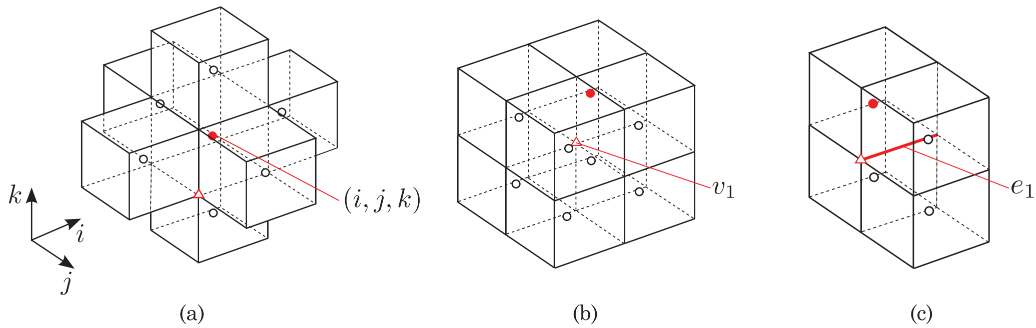


Fig. 2. Sketches of the neighboring voxels adjacent to voxel (i, j, k) used to compute the number of exterior faces (a), vertices (b), and edges (c). In all plots, the closed red dot denotes the center of the voxel (i, j, k) and the triangle the position of its vertex v_1 as defined in Figure 1(b). Case (c) is particularized to edge e_1, and similar configurations apply to edges e_2 and e_3 (see Figure 1(b)).

ALGORITHM 1: Count Number of Exterior Faces at Position (i, j, k).

Input: A,F
Output: F
if *A is equal to 1 at position (i, j, k)* **then**
    if *A is equal to 0 at position (i − 1, j, k)* **then**
        F ← F + 1;
    end
    if *A is equal to 0 at position (i + 1, j, k)* **then**
        F ← F + 1;
    end
    if *A is equal to 0 at position (i, j − 1, k)* **then**
        F ← F + 1;
    end
    if *A is equal to 0 at position (i, j + 1, k)* **then**
        F ← F + 1;
    end
    if *A is equal to 0 at position (i, j, k − 1)* **then**
        F ← F + 1;
    end
    if *A is equal to 0 at position (i, j, k + 1)* **then**
        F ← F + 1;
    end
end

extending it. *cube*1 and *cube*2 are auxiliary arrays of integers with dimensions $2 \times 2 \times 2$ and 2×2, respectively, and are used to store the slices of A shown in Figure 2(b) and (c).

(2) *Loop through* $i = 2, \ldots, N_x - 1$, $j = 2, \ldots, N_y - 1$, $k = 2, \ldots, N_z - 1$. For each (i, j, k), proceed as follows:

(a) *Count number of exterior faces*. See Algorithm 1. The six faces of the voxel at (i, j, k) are considered, and its six neighbors are defined in Figure 2(a). For each neighboring voxel with coordinates $(i + \Delta i, j + \Delta j, k + \Delta k)$, F is increased by one if $A(i, j, k) = 1$ and $A(i + \Delta i, j + \Delta j, k + \Delta k) = 0$. The possible values for $(i+\Delta i, j+\Delta j, k+\Delta k)$ are $(i-1, j, k)$, $(i+1, j, k)$, $(i, j-1, k)$, $(i, j+1, k)$, $(i, j, k-1)$, and $(i - 1, j, k + 1)$.

ALGORITHM 2: Count Number of Exterior Vertices at Position (i, j, k).

Input: A,V
Output: V
cube1 $\leftarrow$ slice of A from $i-1$ to i, $j-1$ to j and $k-1$ to k;
if *any element in cube1 is 0* and *any element in cube1 is 1* **then**
  V $\leftarrow$ V + getconnected1(cube1);
end

ALGORITHM 3: Count Number of Exterior Edges at Position (i, j, k).

Input: A,E
Output: E
cube2 $\leftarrow$ slice of A from $i-1$ to i, $j-1$ to j and k;
if *any element in cube2 th* and *any element in cube2 is 1* **then**
  E $\leftarrow$ E + getconnected2(cube2);
end

(b) *Count number of exterior vertices*. See Algorithm 2. Only vertex v_1 in Figure 1(b) is considered, and its eight adjacent voxels are defined in Figure 2(b). V is increased by one if any of the eight adjacent voxels has value 1, and any other value 0. In some cases, V must increase by a number dV larger than one if some of the surrounding voxels are locally not connected. The value of dV is calculated by procedure *getconnected*1, which is discussed at the end of the section.
(c) *Count number of exterior edges*. See Algorithm 3. Only edges e_1, e_2, and e_3 highlighted in Figure 1(b) are considered. The four adjacent voxels for edge e_1 are shown in Figure 2(c). E is increased by one unit if any of the four adjacent voxels has value 1 and any other value 0. In some cases, the edges have to be counted dE times when the neighboring voxels are not locally connected. The increment dE is computed by procedure *getconnected*2. A similar algorithm applies to the other two edges e_2 and e_3 shown in Figure 1(b).

(3) Finally, the Euler characteristic and the genus are computed as $X = V - E + F$ and $G = (2 - X)/2$.

To complete the description of the algorithm, we now comment on the procedures *getconnected*1 and *getconnected*2. Some edges or vertices has to be counted multiple times to be consistent with the 6-connectivity of the voxels. An example is illustrated in Figure 1(a). In contrast to other works [Toriwaki and Yonekura 2002; Ayala et al. 2012; Cruz and Ayala 2013], this is achieved by counting the number of local objects contained in the slices shown in Figure 2(b) and (c)—that is, the number of objects in the $2 \times 2 \times 2$ subvolume satisfying the 6-connectivity disregarding any other connections outside the slide. For example, the subvolume denoted as C41 in Figure 3 contains one local object, and C33 contains three. Note that some voxels may be locally disconnected but belong to the same object, as they may connect indirectly through other voxels not considered in the slide. The purpose of procedure *getconnected*1, defined in Algorithm 4, is to compute the number of local objects in the subvolume shown in Figure 2(b), which can easily be obtained by any labeling method, such as the Hoshen-Kopelman algorithm [Hoshen and Kopelman 1976]. Note that there is one degenerated case with an infinitesimally small hole, shown in case C63 in Figure 3, where there is only one object but the vertex must be considered twice.

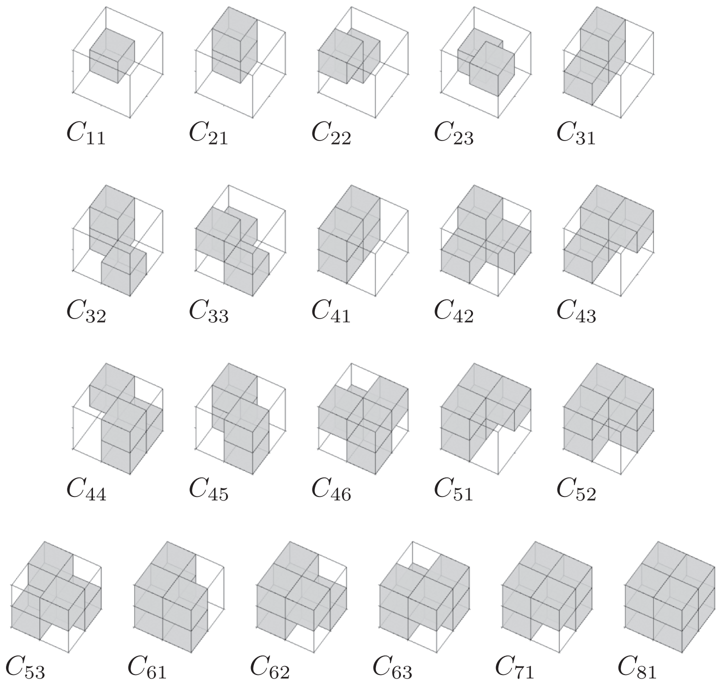


Fig. 3. All possible configurations of voxels in a $2 \times 2 \times 2$ subvolume considering symmetries. Grey cubes represent voxels with value 1. The cases are denoted by C_{ij}, where i is the number of voxels with value 1 in the subvolume.

ALGORITHM 4: Procedure Getconnected1. It Computes the Increment dV for V.

Input: cube1
Output: dV
if *cube1 is degenerated case C63 in Figure 3* **then**
  dV ← 2;
else
  dV ← number of local objects in cube1;
end

ALGORITHM 5: Procedure Getconnected2. It Computes the Increment dE for E.

Input: cube2
Output: dE
if *all the elements equal to 1 in cube2 are locally connected* **then**
  dE ← 1;
else
  dE ← 2;
end

Procedure *getconnected*2 is presented in Algorithm 5 and follows the same idea. In this case, the only possible configuration to obtain more than one local object in the slide shown in Figure 2(c) is with two voxels that do not share any face. In the rest of the cases, the number of local objects is one.

Table I. Contribution to the Number of Faces ΔF, Edges ΔE, and Vertices ΔF of Each Configuration of Voxels in the $2 \times 2 \times 2$ Subdomains Shown in Figure 3

Case	ΔF	ΔE	ΔV	Case	ΔF	ΔE	ΔV
C_{01}	0	0	0	C_{44}	8	8	2
C_{11}	3	3	1	C_{45}	8	8	2
C_{21}	4	4	1	C_{46}	12	12	4
C_{22}	6	6	2	C_{51}	5	5	1
C_{23}	6	6	2	C_{52}	7	7	1
C_{31}	5	5	1	C_{53}	9	9	2
C_{32}	7	7	2	C_{61}	4	4	1
C_{33}	9	9	3	C_{62}	6	6	1
C_{41}	4	4	1	C_{63}	6	6	2
C_{42}	6	6	1	C_{71}	3	3	1
C_{43}	6	6	1	C_{81}	0	0	0

4. ALTERNATIVE ALGORITHM

An alternative algorithm is introduced for the purpose of validating the approach presented earlier. Conceptually, it follows the same ideas discussed in Section 3 but relies on a precomputed table of cases, as in the work by Toriwaki and Yonekura [2002]. The process involves looping through all vertices of the virtual grid, counting vertices, faces, and edges, but no effort is made to prevent multiple counts of the last two, as opposed to the algorithm presented in Section 3. This results in an extra number of faces and edges that is easily corrected by dividing the total number of faces by four and of edges by two, the reason being that each face and edge contains four and two vertices, respectively.

The number of faces and edges at a particular vertex depends on its eight surrounding voxels as shown in Figure 2(b). In this scenario, there are 256 different cases that may be reduced by symmetry to those shown in Figure 3. We will use the index l to sequentially label the vertices of the virtual mesh. The contributions of the l-th vertex to the total number of faces, edges, and vertices will be denoted by ΔF_l, ΔE_l, and ΔV_l, respectively, and their values are tabulated in Table I for all possible cases. F, E, and V are then obtained as

$$F = \frac{\sum_l \Delta F_l}{4}, \quad E = \frac{\sum_l \Delta E_l}{2}, \quad V = \sum_l \Delta V_l, \tag{5}$$

where the summation extends to all vertices of the virtual mesh. Finally, the Euler characteristic and the genus are calculated with (4) and (2), respectively.

The alternative algorithm is now briefly described following the same notation used in the previous section:

(1) *Initialize the variables*. F, V, and E are equivalent to those described in Section 3 and are initialized to zero.
(2) *Loop through all vertices in A.* At the l-th vertex, proceed as follows:
   (a) *Find the case*. Considering the eight surrounding voxels at the l-th vertex shown in Figure 2(b), search for the corresponding case in Figure 3, taking into account symmetries.
   (b) *Contributions to F, E, and V*. From Table I, obtain ΔF_l, ΔE_l, and ΔV_l, and compute $F \leftarrow F + \Delta F_l$, $E \leftarrow E + \Delta E_l$, and $V \leftarrow V + \Delta V_l$.
(3) *Compute the actual number of faces and edges*. $F \leftarrow F/4$ and $E \leftarrow E/2$.
(4) *Compute the Euler characteristic and genus*. $X = V - E + F$ and $G = (2 - X)/2$.

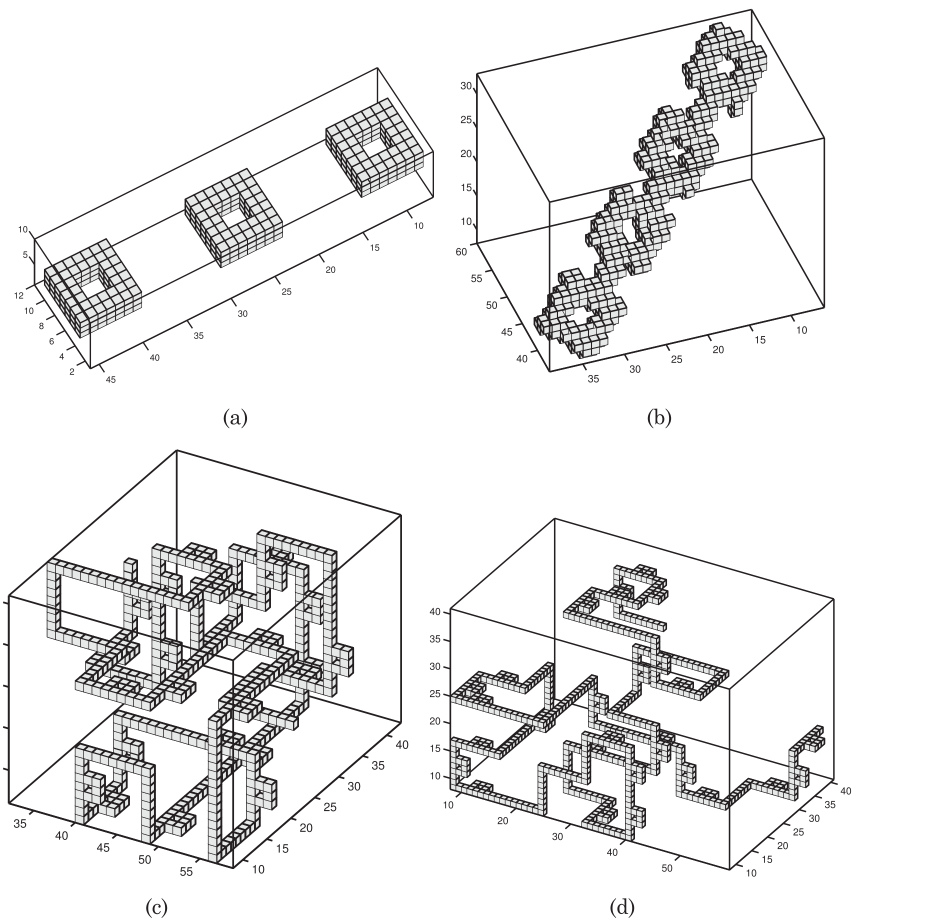


Fig. 4. Examples of synthetic cases used to validate the algorithms. (a) Isolated torus. (b) Concatenated torus. (c, d) Synthetic random cases generated from the building blocks shown in Figure 5.

The algorithm described in Section 3 is between 1.2 and 2 times faster than the one presented in this section, and roughly 3 times shorter in terms of lines of code, which makes it more efficient and simple to implement. For those reasons, the former approach is preferred, and the alternative algorithm is only considered for validation purposes in the next section.

5. VALIDATION AND SCALABILITY

Two approaches are followed to validate the algorithms detailed in Sections 3 and 4. First, synthetic cases whose genus are known beforehand are fed into the algorithms, and the results are compared to the expected theoretical values. Second, different datasets are used to compute the genus with both algorithms, and the outputs are shown to match.

The synthetic cases tested are the following: all possible configurations in a $2 \times 2 \times 2$ volume, n number of isolated solid objects ($g = -n + 1$), n isolated objects with an interior cavity each ($g = -2n + 1$), n isolated torus ($g = 1$) as in the example shown in Figure 4(a), and n torus connected by solid bridges ($g = n$) as in Figure 4(b). More cases were tested by rotating the previous ones at different angles, such as in the case

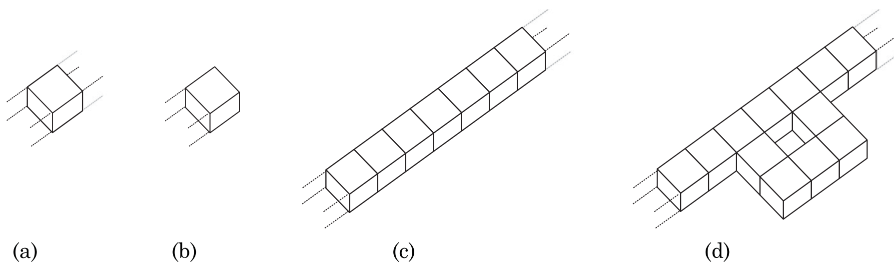


Fig. 5. Building blocks of randomly generated test cases. (a) Node. (b) End. (c) Connector type I. (d) Connector type II.

Table II. Contribution to the Number of Faces ΔF, Edges ΔE, and Vertices ΔV of the Different Blocks Shown in Figure 5

Case	End	Node	Connector Type I	Connector Type II
ΔF	5	4	28	46
ΔE	12	12	52	88
ΔV	8	8	24	40

shown in Figure 4(b). The values of n tested range from 1 to 10^6. One more synthetic case tested consists of randomly generated structures built using the blocks shown in Figure 5, referred to as nodes and ends and linked by two type of connectors. The number of faces, edges, and vertices of the resulting object is given by

$$F = n_e \Delta F_e + n_n \Delta F_n + n_I \Delta F_I + n_{II} \Delta F_{II}, \tag{6}$$

$$E = n_e \Delta E_e + n_n \Delta E_n + n_I \Delta E_I + n_{II} \Delta E_{II}, \tag{7}$$

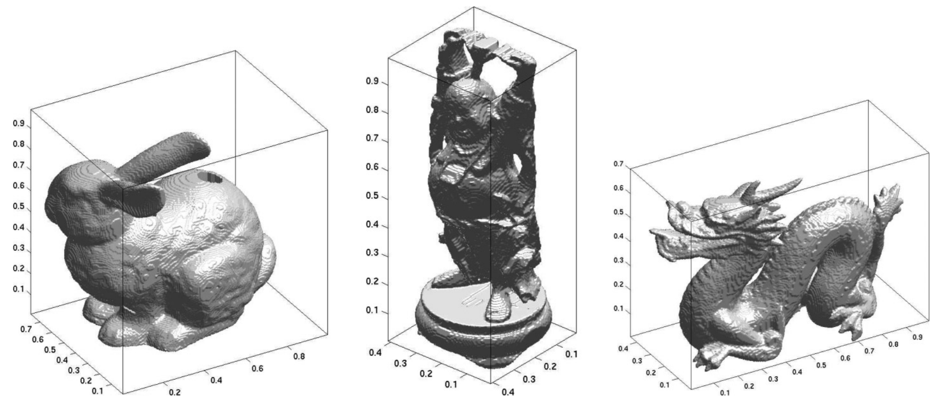
$$V = n_e \Delta V_e + n_n \Delta V_n + n_I \Delta V_I + n_{II} \Delta V_{II}, \tag{8}$$

where n_e, n_n, n_I, and n_{II} are the number of ends, nodes, and connectors of type I and II that belong to the object. The increments ΔF_i, ΔE_i, and ΔV_i with $i = e, b, I$, and II are the contribution to the number of faces, edges, and vertices of each block, respectively, and its values are tabulated in Table II. Roughly 10^6 cases were randomly generated and tested, and two examples are shown in Figure 4(c) and (d). More synthetic cases similar to those presented earlier but using differently shaped connectors were also successfully tested (not shown).

We perform a second validation comparing the number of faces, edges, and vertices computed with the algorithm presented in Section 3 and the alternative one in Section 4, which of course must be identical. This was verified for the synthetic cases described earlier. More test cases are the three models from the Stanford 3D Scanning Repository [Stanford 2014] voxelized with *binvox* [Min 2015] (see also Nooruddin and Turk [2003]) and shown in Figure 6. Finally, we tested 10^2 cases delimited by a cubical region and with grid sizes from 16^3 up to $4,096^3$ whose voxels were randomly initialized with zeros and ones filling approximately 50% of the total volume. Two examples are shown in Figure 7. The two algorithms yield identical results for all cases tested, counting exactly the same number of faces, edges, and vertices and, therefore, the same genus.

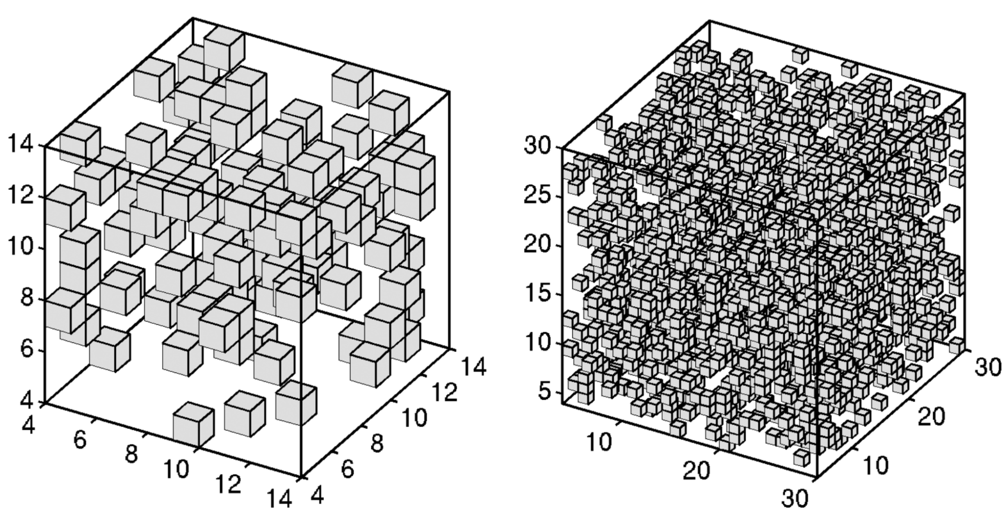
Table III summarizes the number of voxels, faces, edges, vertices, and genus of some of the cases tested, which are available for download in our Web page [Computational Fluid Mechanics Lab 2015].

The algorithm presented in Section 3 was implemented in Fortran, compiled with Intel Fortran Studio XE 2016 16.0.0 20150815, and tested in an Intel® Xeon® CPU X5650 2.67GHz with 192GiB of RAM for cubical arrays of different sizes and randomly



(a) (b) (c)

Fig. 6. Three-dimensional models obtained from the Stanford 3D Scanning Repository and voxelized with binvox. (a) Bunny [Turk and Levoy 1994]. (b) Buda [Curless and Levoy 1996]. (c) Dragon [Curless and Levoy 1996].



(a) (b)

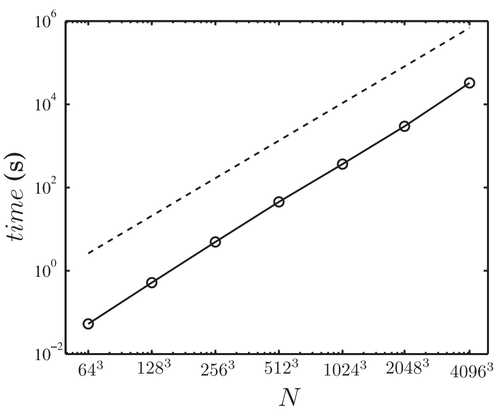
Fig. 7. Test cases of cubical domains with 16^3 (a) and 32^3 (b) voxels randomly initialized with zeros and ones.

generated as those shown in Table III. The average time elapsed to compute the genus of inputs with different sizes is presented in Figure 8(a), which shows linear scalability and makes feasible applications to very large datasets.

The code was also parallelized using Fortran coarrays. The domain decomposition was performed by dividing the z direction in chunks of size $N_x \times N_y \times \Delta N_z$, where ΔN_z is N_z/n_{proc} rounded to the nearest whole number, and n_{proc} the number of processing elements. Two overlapped x-y planes are added at the beginning and end of each chunk to compute faces, edges, and vertices without any extra communication between images. Once this is done, the genus is obtained by summing the faces, edges, and vertices of all chunks. The parallelization works for any number of processing elements smaller than N_z, and the size of the last chunk may differ from the size of the others

Table III. Summary of Some of the Datasets Tested and Available for Download at the Computational Fluid Mechanics Lab [2015]

Case	Size	Faces	Edges	Vertices	Genus
Synthetic1	64^3	1,924	3,848	1,880	23
Synthetic2	64^3	2,174	4,348	2,120	28
Bunny	256^3	309,482	618,964	309,466	9
Buda	256^3	129,800	259,600	129,780	11
Dragon	256^3	164,494	328,988	164,494	1
Random1	64^3	297,496	594,992	280,160	8,669
Random2	128^3	2,744,830	5,489,660	2,570,182	87,325
Random3	256^3	23,530,742	47,061,484	21,985,520	772,612
Random4	512^3	194,709,102	389,418,204	181,726,644	6,491,230
Random5	$1,024^3$	1,584,014,008	3,168,028,016	1,477,589,086	53,212,462
Random6	$2,048^3$	12,778,133,206	25,556,266,412	11,916,193,918	430,969,645
Random7	$4,096^3$	102,651,228,492	205,302,456,984	95,713,851,166	3,468,688,664



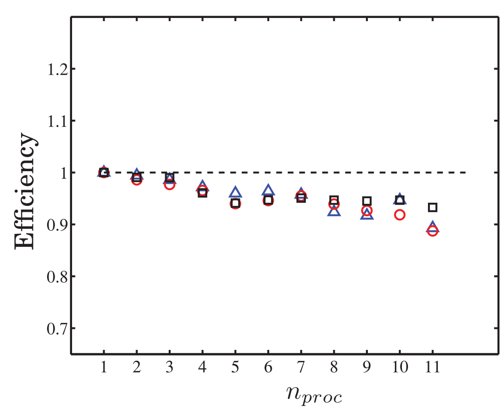


Fig. 8. (a) For the Fortran serial version, average time in seconds elapsed to compute the genus of cubical arrays of size N^3 randomly initialized to zeros and ones with roughly 50% of the volume occupied. Results for a single processor. The circles are the measured times, and the solid dashed line is $time \sim N$. (b) For the Fortran coarrays version, strong scaling efficiency as a function of the number of processing elements, n_{proc}, for three different problem sizes: $N = 1024^3$ ($\triangle$), $N = 2048^3$ ($\circ$), and $N = 4096^3$ ($\square$).

if N_z is not divisible by n_{proc}. The reader is referred to the software component of the manuscript to cover all details of the parallelization. The strong scaling efficiency, where the problem size stays fixed but the number of processing elements increases, is shown in Figure 8(b). The results are quite satisfactory, and the efficiency always remains greater than 90%.

6. APPLICATIONS TO TURBULENT FLOWS

We show two examples where the genus is used as a tool to characterize the topology of regions of interest in turbulent flows. In the first example, the genus is computed for millions of individual coherent structures extracted from a turbulent channel flow. In the second one, the genus is used to identify physically meaningful interfaces separating turbulent and nonturbulent flow in a time-decaying jet.

6.1. Topology of Coherent Regions in Turbulent Flows

We use three direct numerical simulations of turbulent channel flows (two parallel walls delimiting a flow moving on average in one direction) from Lozano-Durán and

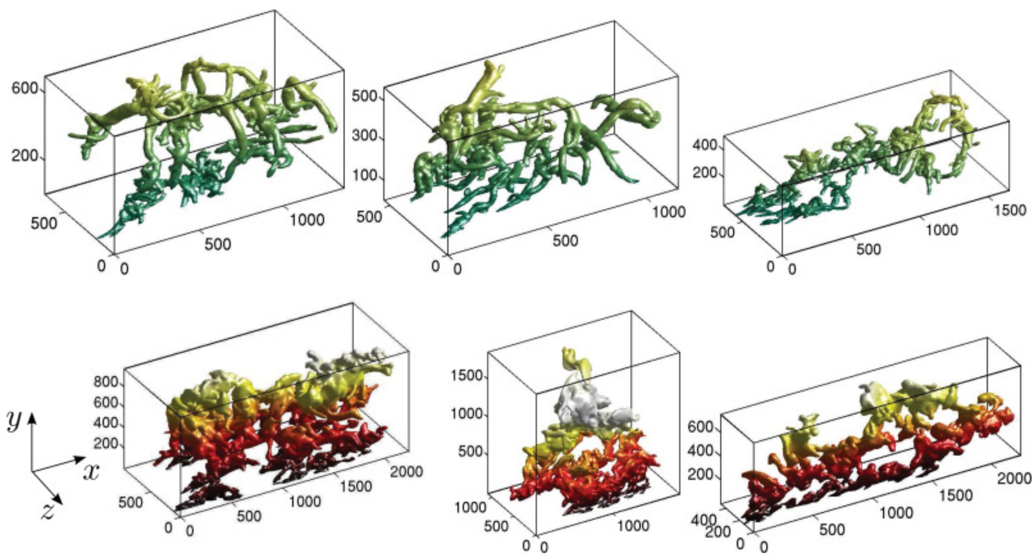


Fig. 9. Examples of three vortex clusters (top row) and Q-structures (bottom row) extracted from a direct numerical simulation of a turbulent channel at $Re_\tau = 4{,}180$ [Lozano-Durán and Jiménez 2014]. The flow goes from bottom left to top right. The axes are normalized with ν/u_τ. The colors change gradually with the distance to the wall, which is located at $y = 0$. Note that the objects are not to scale.

Jiménez [2014] at Reynolds numbers $Re_\tau = 934, 2,004$, and $4,180$, with $Re_\tau = hu_\tau/\nu$, where h is the channel half-height, u_τ the friction velocity, and ν the kinematic viscosity. More details about turbulent channel flows may be found in Chapter 7.1 of Pope [2000]. The streamwise, wall-normal, and spanwise directions are denoted by x, y, and z, respectively. Very briefly, we compute the genus of coherent structures, namely regions of the flow where a variable is higher than a prescribed threshold. The three-dimensional coherent structures under study are vortex clusters from del Álamo et al. [2006] and Q-structures from Lozano-Durán et al. [2012]. The former are defined in terms of the discriminant of the velocity gradient and are connected regions satisfying

$$D(x, y, z)/D'(y) > \alpha, \tag{9}$$

where D is the instantaneous discriminant of the velocity gradient tensor, $D'(y)$ its standard deviation at each $x-z$ plane, and $\alpha = 0.02$ a thresholding parameter obtained from a percolation analysis. Similarly, Q-structures are defined as places where

$$uv(x, y, z)/uv'(y) > H, \tag{10}$$

where uv is the instantaneous tangential Reynolds stress, being u and v the streamwise and wall-normal velocity fluctuations, $uv'(y)$ its rooted-mean-squared value at each y-position, and H a thresholding parameter equal to 1.75. Three-dimensional objects are constructed by connecting neighboring grid points fulfilling relations (9) for vortex clusters and (10) for Q-structures and using the 6-connectivity criteria. Full details for both types of structures can be found in del Álamo et al. [2006], Lozano-Durán et al. [2012], and Lozano-Durán and Jiménez [2014]. To compute the genus, each object is circumscribed within a box aligned to the Cartesian axes, which constitutes the limits of the array $A(i, j, k)$ discussed in Section 3. Figure 9 shows several examples of actual objects extracted from the flow and demonstrates the complex geometries that

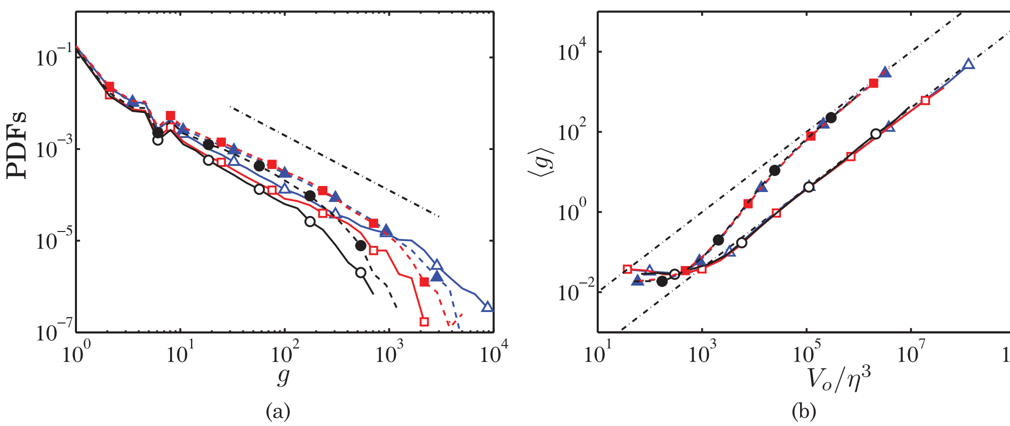


Fig. 10. (a) PDFs of the genus. The dashed-dotted line is proportional to $g^{-1.2}$. (b) Average number of holes (genus) of individual coherent structures as a function of their volume, V_o, in Kolmogorov units. The dashed-dotted lines are $\langle g\rangle = 10^{-3}\eta^{-3}(V_o/\eta^3)$ and $\langle g\rangle = 4\times 10^{-5}\eta^{-3}(V_o/\eta^3)$. For (a) and (b), the solid lines with open symbols correspond to Q-structures, and the dashed lines with closed symbols correspond to vortex clusters. Different symbols stand for different Reynolds numbers: ○, $Re_\tau = 934$; □, $Re_\tau = 2,004$; △, and $Re_\tau = 4{,}180$.

may appear. The number of structures computed is of the order of 10^7, with a wide spectrum of sizes ranging from $\sim 30^3$ to $\sim 2,000^3$ voxels.

Each array $A(i, j, k)$ contains just one single object, and hence the only contributions to the genus are the number of holes and internal cavities. The data reveals that only 0.05% of objects have negative genus, and it was checked that most of structures are solid. In this scenario, the genus and number of holes can be used interchangeably.

The probability density functions (PDFs) of the genus, g, are presented in Figure 10(a), and most of the values concentrate around zero or a few holes, although the long potential tails reach values up to 10^4 holes. Figure 10(b) shows the average number of holes in the objects as a function of their volume, V_o, normalized in Kolmogorov units, η^3 (see Chapter 7 of Pope [2000]). It becomes clear that as the volume of the structures increases, so does the genus, which is reasonable if we consider that the volume of the object is related to its internal Reynolds number (or complexity), and increasing its volume results in more complicated topologies. The curves for both vortex clusters and Q-structures show good collapse for the three Reynolds numbers and follow the trend $\langle g\rangle = \rho V_o$, with $\langle g\rangle$ the average genus for a given volume and ρ a constant equal to $10^{-3}\eta^{-3}$ and $4\times 10^{-5}\eta^{-3}$ for vortex clusters and Q-structures, respectively.

From relation $\langle g\rangle = \rho V_o$, the genus may be understood as an alternative method to characterize the level of complexity of the structures, with ρ a density equal to the number of holes per unit volume. If we define l as the average distance between holes within the structures, its value may be approximated as $l \approx (10^{-3})^{-1/\alpha_c}\eta \approx 30\eta$ for vortex clusters and $l \approx (4\times 10^{-5})^{-1/\alpha_Q}\eta \approx 90\eta$ for Q-structures, with $\alpha_c = 2$ and $\alpha_Q = 2.25$ the average fractal dimensions of the objects computed by Lozano-Durán et al. [2012]. These lengths are consistent with a model of coherent structures built by small blocks of length 30 to 90η stacked together to create larger objects but not perfectly compacted, which results in holes between the blocks. For a given volume, V_o, vortex clusters have an average of 25 times more holes than Q-structures, suggesting that their blocks and connections are fundamentally different. This is consistent with Lozano-Durán et al. [2012], who showed that the Q-structures are flake shaped, whereas vortex clusters are worm shaped (also visible in Figure 9).

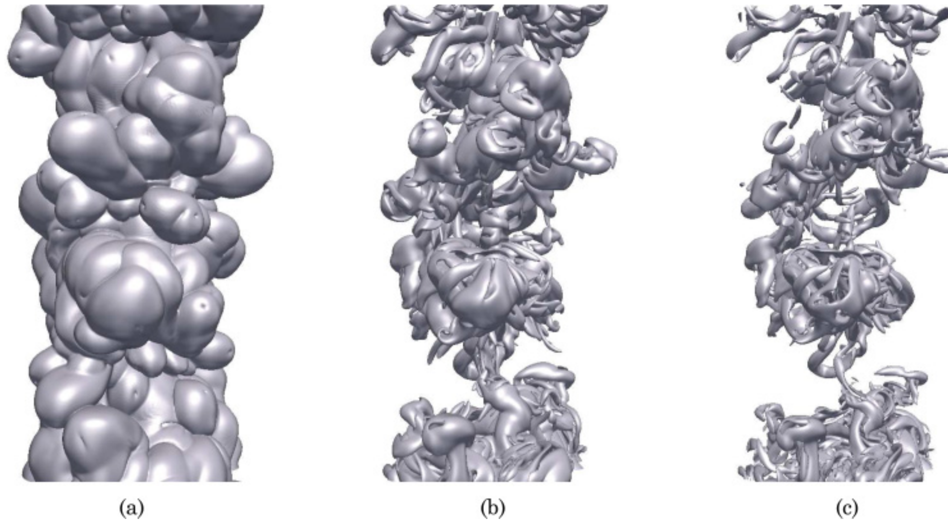


Fig. 11. Isocontours of vorticity magnitude $|\omega|$ at different thresholds for a turbulent jet. Part (a) corresponds to a very low value of the threshold $|\omega|/\omega_{rms} = 0.05$, and very low genus, where ω_{rms} is the rooted-mean-squared vorticity magnitude. Part (b) corresponds to the threshold that maximize the genus $|\omega|/\omega_{rms} = 3$. Part (c) corresponds to a threshold slightly higher than (b), $|\omega|/\omega_{rms} = 5$.

6.2. Turbulent/Nonturbulent Interface Detection in a Turbulent Jet

We use a direct numerical simulation of a time-decaying turbulent jet (see Chapter 5 in Pope [2000]) by Vela-Martín and Borrell [2014] to identify a turbulent/nonturbulent interface. A brief introduction about such an interface is presented next.

Two regions can be distinguished in an unbounded turbulent flow, the fully turbulent region, characterized by strong fluctuations, and the irrotational free stream. These two regions are in most cases separated by a single thin layer, called the *turbulent/nonturbulent interface*. The first step to analyze the physical processes that happen within this interface layer is to locate it. This interface is known to be fractal like [Sreenivasan et al. 1989], and it contains all of the scales between the smallest and the largest possible. Such a wide range of scales imposes a strong restriction on the size of the domain that has to be studied, as small portions would only give reliable results for the small scales. The most common method to locate the turbulent/nonturbulent interface is to threshold a scalar field where the two characteristic states of the flow can easily be distinguished. Sreenivasan et al. [1989] and Westerweel et al. [2009] use the concentration of a passive scalar injected in the turbulent side and threshold it at the least probable value of the concentration. Bisset et al. [2002] and da Silva and Taveira [2010] use a particular isocontour of the magnitude of vorticity $|\omega|(x, y, z) = \omega_0$, where vorticity is defined as the rotational of the velocity vector, $\vec{\omega} = \nabla \wedge \vec{u}$. Gampert et al. [2014] found that the isocontours obtained thresholding concentration and vorticity magnitude are similar, and da Silva et al. [2014b] found that the least probable value of vorticity magnitude can be used successfully as a threshold for a variety of turbulent flows. Despite the convergence of some popular methodologies, other authors, such as Chauhan et al. [2014] have proposed alternative strategies.

One important aspect of the choice of the threshold is the impact it has on the geometry of the interface. If the threshold ω_0 is a low value of vorticity, such as the detection shown in Figure 11(a), the interface is relatively simple, showing that the perturbation caused by the turbulent motion is smoothed out further down the free stream. On the other hand, as soon as the threshold is slightly increased, the surface is populated

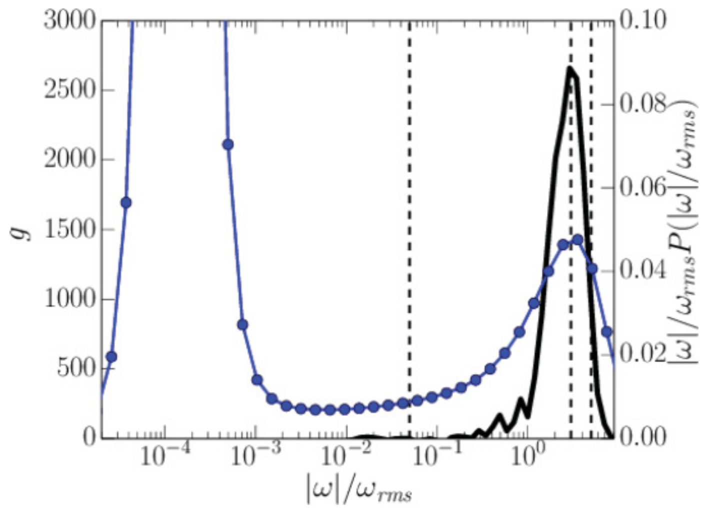


Fig. 12. The blue line with circles shows a premultiplied PDF of vorticity magnitude $|\omega|$ normalized with its root-mean-squared value in a turbulent temporal round jet. The solid black line shows the genus of the turbulent/nonturbulent interface as a function of the vorticity magnitude. The vertical dashed lines are the thresholds used in Figure 11(a) through (c).

with a large amount of handles (or holes), as can be seen in Figure 11(b) and (c). These handles are most likely a geometrical feature of the fully turbulent flow. Depending on the value of the threshold, the surface generated has different topological properties.

The geometrical complexity, measured in this case with the number of handles, has an important side effect on the analysis of the properties of the flow depending on the relative position to the interface. Two relatively popular assumptions about the interface are that there is a privileged direction across which the relative distance to the interface can be measured [Westerweel et al. 2009; da Silva and Taveira 2010], and that the interface is simple enough so that a local normal is meaningful [Bisset et al. 2002; Chauhan et al. 2014]. These two assumptions are not strictly correct if handles are a dominant feature of the interface. At the same time, the criterion explored by da Silva et al. [2014b] depends on the characteristics of the nonturbulent region. The PDF of vorticity in the same round jet of Figure 11 is shown in Figure 12. It has been premultiplied to emphasize the fact that the PDF has two major contributions: one from the bulk of the nonturbulent flow with low vorticity (left peak) and a second one from the bulk of the turbulent flow with high vorticity (right peak). Note that if the flow was in an ideal state with no perturbations, the left peak would be in the limit of vanishing vorticity. As a consequence, the outcome of the criterion defined by Sreenivasan et al. [1989] applied to the vorticity field can be intuitively defined as the lowest threshold that is not affected by the spurious vorticity present in the free stream. This criterion is strictly correct, but it may be more representative of the smoothed-out perturbations relatively far from the turbulent motion. Therefore, it is necessary to explore other complementary threshold choices that provide a more complete description of the vorticity field.

The genus of the surface detected as a function of the value used to threshold the vorticity magnitude is presented in Figure 12. The results have been averaged using an ensemble of four equivalent cases. The curve shows that there is a gradual yet evident change in the topological properties of the vorticity interface from a threshold $\omega_0 \sim \omega_{rms}$. Beyond that value, handles are a dominant feature of the interface, and the standard tools for the conditional analysis are probably not valid. If the criterion of minimum probability provides a lower limit for the threshold, the genus of the interface is an useful criterion for an upper limit.

7. CONCLUSIONS

We have presented and validated a simple algorithm to numerically compute the genus of discrete surfaces using the Euler characteristic formula. The method is valid for surfaces associated with three-dimensional objects obtained by thresholding a discrete scalar field defined in a structured-collocated grid and offers several advantages. First, it does not rely on any direct triangulation of the surfaces, which is usually memory- and time consuming. In addition, the surfaces of all three-dimensional objects in the domain are automatically detected, and the genus is exactly computed without any spurious holes. Last but not least, it needs practically zero memory, it is fast and scalable, and it has a computational cost directly proportional to the size of the grid computed. The algorithm is also highly parallelizable, and a Fortran coarrays version was implemented to take advantage of multicore processors without increasing the memory usage. This makes the algorithm suitable for large datasets, such as the ones encountered in direct numerical simulations of turbulent flows. Two applications to the characterization of complex structures in turbulent flows have been presented. In the first case, the genus of coherent structures extracted from a turbulent channel flows is computed and found to be proportional to the volume of the objects. In the second application, the genus is used to find an appropriate threshold to detect the turbulent/nonturbulent interface in a turbulent jet.

ACKNOWLEDGMENTS

The authors would like to acknowledge fruitful discussions with Javier Jiménez and José Cardesa-Dueñas, and to thank Professors Dolors Ayala and Irving Cruz for providing the test data used in the early stages of the work. We are also very grateful to Tim Hopkins for his assistance with the software component of the article and to the referees for their very constructive feedback.

REFERENCES

Dolors Ayala, Eduard Vergés, and Irving Cruz. 2012. A polyhedral approach to compute the genus of a volume dataset. In *Proceedings of the International Conference on Computer Graphic Theory and Applications (GRAPP'12)*. 38–47.

D. K. Bisset, J. C. R. Hunt, and M. M. Rogers. 2002. The turbulent/non-turbulent interface bounding a far wake. *Journal of Fluid Mechanics* 451, 383–410. DOI:http://dx.doi.org/10.1017/S0022112001006759

G. Borrell and J. Jiménez. 2013. Geometrical properties and scaling of the turbulent-nonturbulent interface in boundary layers. In *Proceedings of the 66th Annual Meeting of the APS Division of Fluid Dynamics*. http://adsabs.harvard.edu/abs/2013APS..DFDR31002B.

Kapil Chauhan, Jimmy Philip, Charitha M. de Silva, Nicholas Hutchins, and Ivan Marusic. 2014. The turbulent/non-turbulent interface and entrainment in a boundary layer. *Journal of Fluid Mechanics* 742, 119–151. DOI:http://dx.doi.org/10.1017/jfm.2013.641

Isaac Chavel. 2006. *Riemannian Geometry* (2nd ed.). Cambridge University Press, Cambridge, UK.

Li Chen and Yongwu Rong. 2010. Digital topological method for computing genus and the Betti numbers. *Topology and Its Applications* 157, 12, 1931–1936. DOI:http://dx.doi.org/10.1016/j.topol.2010.04.006

Computational Fluid Mechanics Lab. 2015. Index of /Genus. Retrieved May 20, 2016, from http://torroja.dmt.upm.es/genus/.

Irving Cruz and Dolors Ayala. 2013. An efficient alternative to compute the genus of binary volume models. In *Proceedings of the International Conference on Computer Graphics Theory and Applications (GRAPP'13)*. 18–26.

Brian Curless and Marc Levoy. 1996. A volumetric method for building complex models from range images. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'96)*. ACM, New York, NY, 303–312. DOI:http://dx.doi.org/10.1145/237170.237269

Carlos B. da Silva, Julian C. R. Hunt, Ian Eames, and Jerry Westerweel. 2014a. Interfacial layers between regions of different turbulence intensity. *Annual Review of Fluid Mechanics* 46, 1, 567–590. DOI:http://dx.doi.org/10.1146/annurev-fluid-010313-141357

Carlos B. da Silva and Rodrigo R. Taveira. 2010. The thickness of the turbulent/nonturbulent interface is equal to the radius of the large vorticity structures near the edge of the shear layer. *Physics of Fluids* 22, 121702.

Carlos B. da Silva, Rodrigo R. Taveira, and Guillem Borrell. 2014b. Characteristics of the turbulent/nonturbulent interface in boundary layers, jets and shear-free turbulence. *Journal of Physics: Conference Series* 506, Article No. 15.

Juan C. del Álamo, Javier Jiménez, Paulo Zandonade, and Robert D. Moser. 2006. Self-similar vortex clusters in the turbulent logarithmic region. *Journal of Fluid Mechanics*, 329–358. DOI:http://dx.doi.org/10.1017/S0022112006000814

J. Einasto, M. Einasto, E. Tago, E. Saar, G. Htsi, M. Jeveer, L. J. Liivamgi, et al. 2007. Superclusters of galaxies from the 2D Redshift survey. *Astronomy and Astrophysics* 462, 2, 811–825. DOI:http://dx.doi.org/10.1051/0004-6361:20065296

Markus Gampert, Jonas Boschung, Fabian Hennig, Michael Gauding, and Norbert Peters. 2014. The vorticity versus the scalar criterion for the detection of the turbulent/non-turbulent interface. *Journal of Fluid Mechanics* 750, 578–596.

J. R. Gott III, M. Dickinson, and A. L. Melott. 1986. The sponge-like topology of large-scale structure in the universe. *Astrophysical Journal* 306, 341–357. DOI:http://dx.doi.org/10.1086/164347

J. R. Gott III, J. Miller, T. X. Thuan, S. E. Schneider, D. H. Weinberg, C. Gammie, K. Polk, et al. 1989. The topology of large-scale structure. III—Analysis of observations. *Astrophysical Journal* 340, 625–646. DOI:http://dx.doi.org/10.1086/167425

J. R. Gott III, D. H. Weinberg, and A. L. Melott. 1987. A quantitative approach to the topology of large-scale structure. *Astrophysical Journal* 319, 1–8. DOI:http://dx.doi.org/10.1086/165427

Alan H. Guth. 1981. Inflationary universe: A possible solution to the horizon and flatness problems. *Physical Review D* 23, 2, 347–356. DOI:http://dx.doi.org/10.1103/PhysRevD.23.347

A. J. S. Hamilton, J. R. Gott III, and D. Weinberg. 1986. The topology of the large-scale structure of the universe. *Astrophysical Journal* 309, 1–12. DOI:http://dx.doi.org/10.1086/164571

J. Hoshen and R. Kopelman. 1976. Percolation and cluster distribution. I. Cluster multiple labeling technique and critical concentration algorithm. *Physical Review B* 14, 8, 3438–3445. DOI:http://dx.doi.org/10.1103/PhysRevB.14.3438

Sergio Hoyas and Javier Jiménez. 2008. Reynolds number effects on the Reynolds-stress budgets in turbulent channels. *Physics of Fluids* 20, 10, 101511. DOI:http://dx.doi.org/10.1063/1.3005862

Yukio Kaneda, Takashi Ishihara, Mitsuo Yokokawa, Kenichi Itakura, and Atsuya Uno. 2003. Energy dissipation rate and energy spectrum in high resolution direct numerical simulations of turbulence in a periodic box. *Physics of Fluids* 15, 2, L21–L24. DOI:http://dx.doi.org/10.1063/1.1539855

S. E. Konkle, P. Moran, B. Hamann, and K. Joy. 2003. *Fast Methods for Computing Isosurface Topology with Betti Numbers*. Kluwer Academic, Norwell, MA, 363–375.

T. Leung, N. Swaminathan, and P. A. Davidson. 2012. Geometry and interaction of structures in homogeneous isotropic turbulence. *Journal of Fluid Mechanics* 710, 453–481. DOI:http://dx.doi.org/10.1017/jfm.2012.373

A. D. Linde. 1983. Chaotic inflation. *Physical Letters* 129, 34, 177–181. DOI:http://dx.doi.org/10.1016/0370-2693(83)90837-7

Adrián Lozano-Durán, Oscar Flores, and Javier Jiménez. 2012. The three-dimensional structure of momentum transfer in turbulent channels. *Journal of Fluid Mechanics* 694, 100–130. DOI:http://dx.doi.org/10.1017/ jfm.2011.524

Adrián Lozano-Durán and Javier Jiménez. 2014. Effect of the computational domain on direct simulations of turbulent channels up to $Re_\tau = 4200$. *Physics of Fluids* 26, 1, 7. DOI:http://dx.doi.org/10.1063/1.4862918

E. Martin-Badosa, A. Elmoutaouakkil, S. Nuzzo, D. Amblard, L. Vico, and F. Peyrin. 2003. A method for the automatic characterization of bone architecture in 3D mice microtomographic images. *Computerized Medical Imaging and Graphics* 27, 6, 447–458.

K. R. Mecke, T. Buchert, and H. Wagner. 1994. Robust morphological measures for large-scale structure in the universe. *Astronomy and Astrophysics* 288, 697–704.

Patrick Min. 2015. Binvox: Mesh Voxelizer. Retrieved May 20, 2016, from http://www.cs.princeton.edu/∼min/binvox/.

F. S. Nooruddin and G. Turk. 2003. Simplification and repair of polygonal models using volumetric techniques. *IEEE Transactions on Visualization and Computer Graphics* 9, 2, 191–205. DOI:http://dx.doi.org/10.1109/TVCG.2003.1196006

A. Odgaard and H. J. G. Gundersen. 1993. Quantification of connectivity in cancellous bone, with special emphasis on 3-D reconstructions. *Bone* 14, 2, 173–182. DOI:http://dx.doi.org/10.1016/8756-3282(93)90245-6

C. Park, Y.-Y. Choi, M. S. Vogeley, J. R. Gott III, J. Kim, C. Hikage, T. Matsubara, et al. 2005a. Topology analysis of the Sloan Digital Sky Survey. I. Scale and luminosity dependence. *Astrophysical Journal* 633, 11–22. DOI:http://dx.doi.org/10.1086/452625

C. Park, J. Kim, and J. R. Gott III. 2005b. Effects of gravitational evolution, biasing, and Redshift space distortion on topology. *Astrophysical Journal* 633, 1–10. DOI:http://dx.doi.org/10.1086/452621

S. B. Pope. 2000. *Turbulent Flows*. Cambridge University Press, Cambridge, UK.

V. Sahni, B. S. Sathyaprakash, and S. F. Shandarin. 1998. Shapefinders: A new shape diagnostic for large-scale structure. *Astrophysical Journal* 495, 1, L5.

Juan A. Sillero, Javier Jiménez, and Robert D. Moser. 2013. One-point statistics for turbulent wall-bounded flows at Reynolds numbers up to δ+2000. *Physics of Fluids* 25, 10, 105102.

K. R. Sreenivasan, R. Ramshankar, and C. Meneveau. 1989. Mixing, entrainment and fractal dimensions of surfaces in turbulent flows. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences* 421, 1860, 79–108. DOI:http://dx.doi.org/10.1098/rspa.1989.0004

Stanford. 2014. The Stanford 3D Scanning Repository. Retrieved May 20, 2016, from https://graphics.stanford.edu/data/3Dscanrep/.

Anthony C. Thompson. 1996. *Minkowski Geometry*. Cambridge University Press, Cambridge, UK.

J. Toriwaki and T. Yonekura. 2002. Euler number and connectivity indexes of a three dimensional digital picture. *Forma* 17, 183–209.

Greg Turk and Marc Levoy. 1994. Zippered polygon meshes from range images. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'94)*. ACM, New York, NY, 311–318. DOI:http://dx.doi.org/10.1145/192161.192241

Alberto Vela-Martín and Guillem Borrell. 2014. *Computation of a Temporal Decaying Turbulent Jet with GPGPUS*. Technical Report. Computational Fluid Mechanics Group, UPM.

M. S. Vogeley, C. Park, M. J. Geller, J. P. Huchra, and J. R. Gott III. 1994. Topological analysis of the CfA Redshift survey. *Astrophysical Journal* 420, 525–544. DOI:http://dx.doi.org/10.1086/173583

D. H. Weinberg. 1988. Contour: a topological analysis program. *Publications of the Astronomical Society of the Pacific* 100, 1373–1385. DOI:http://dx.doi.org/10.1086/132337

J. Westerweel, C. Fukushima, J. M. Pedersen, and J. C. R. Hunt. 2009. Momentum and scalar transport at the turbulent/non-turbulent interface of a jet. *Journal of Fluid Mechanics* 631, 199–230. DOI:http://dx.doi.org/10.1017/S0022112009006600

Received April 2015; revised October 2015; accepted November 2015