Estimation of Euler Characteristic from Point Data

Daniel A. Klain, Konstantin Rybnikov, Karen Daniels,
Bradford Jones, and Cristina Neacsu
Departments of Computer Science and Mathematical Sciences
University of Massachusetts Lowell, Lowell, MA 01854 USA
dklain@cs.uml.edu krybniko@cs.uml.edu kdaniels@cs.uml.edu

Determination of the geometry and topology of a 3-dimensional body is an important problem appearing in Computer Aided Design, medical tomography, crystallography, molecular biology, etc. The methods used to address this problem depend on the form of input data. Let S be a finite set of sample points from a 3-dimensional body $K \subset \mathbb{R}^3$. In applications one may need to estimate certain *integral properties* of K, like Euler characteristic, surface area, volume, etc. Sometimes a reconstruction of K can be attempted and many reconstruction methods have been suggested – e.g., methods based on Fourier analysis, methods based on weighted Delaunay or Voronoi complexes, such as e.g. the α-shapes method of Edelsbrunner [Ede99], etc. However, if the number of sample points is very large, a piecewise-linear reconstruction is not always possible – for example, at present it is impossible to use any methods based on Delaunay complexes when the number of sample points is about 2^{24} or larger. On the other hand, even if a certain reconstruction method is applicable, naturally appears the question of verification of topological properties of a reconstruction. Most reconstruction methods yield either a simplicial complex or, more generally, a semialgebraic, or even semianalytic set. How well does this new body represent the shape of the original point set S? While S itself has very simple topology, being a finite set, we may think of S as a uniform, Poisson, or lattice sampling from some 3-dimensional body with a well-defined shape, whose intrinsic topology we seek to determine.

Using techniques from integral geometry (see [KR97, San76, Vir88]) we have designed a *numerical estimator* for Euler characteristic, a topological invariant of the ideal reconstruction. Our method, unlike, e.g. the method of α-shapes is entirely *meshless*, that is we do not construct a simplicial (or cell-) complex, nor we try to produce a set of spline surfaces approximating the boundary of K. From the combinatorial complexity point of view our algorithms are simpler than those that attempt to approximate the body with a simplicial complex, since ours require much less working memory (a small fraction of the input size, in practice) and do not maintain large data-structures, such as Delaunay complexes. On the other hand, our method is more computationally robust than methods based on Fourier analysis, since we do not evaluate transcendental functions. Yet another advantage of our approach is that it is robust with respect to floating point errors – for example, the Euler characteristic computation in any dimension evaluates only predicates of degree 2. Finally one of our algorithms handles the case of general dimension in much the same way as it does $\mathbb{R}^3$.

Our methods appear to be very practical. *For example, for a 2^{20} lattice point sampling from a 3D body K of complicated topology (Euler characteristic $|\chi(K)| > 100$) with the local feature size (see Section 5.1) at least 4 times the length of the minimal vector of the lattice we can find the exact value of $\chi(K)$ in a mere 7 seconds on an average 32-bit 2.8GHz Xeon workstation.* To compare, Dey, Giesen and Goswami [DGG03] report that, on a 755MHz Pentium III with 512MB of RAM, reconstruction of a shape from a sampling of 2^{20} points takes 53 minutes. The fastest code that we know of (as of July 2005) for construction of Delaunay triangulations finds Delaunay triangulation for a uniformly distributed set of 2^{20} points in a unit cube in about 200 seconds on hardware and software practically identical to ours (see [BBCK03]).

For uniform and Poisson data our methods require more runs, but still, *for a 100,000 uniform point sampling, with density 100 or larger, from a 3D body with local feature size 1 our algorithm produces a reliable estimate of $\chi(K)$ in 9-10 seconds* (on the same hardware as above). The construction of Delaunay triangulation, a necessary *first* step in all algorithms attempting to reconstruct a body from a point sampling, on faster hardware takes at least 1 minute (Dey, Giesen and Hudson *et al.* report 70.5 sec for 120,000 points and 53 sec for 95,000 [DGH01]).

While Section 1 provides a brief theoretical description of the Euler characteristic, more detailed treatments can be found in [KR97, Mun84, Vir88]. Section 1.0.2 presents a recursive technique for estimating the Euler characteristic of a point set in $\mathbb{R}^n$ using projections of points onto hyperplanes as well as the inclusion-exclusion and homotopy-invariance properties of the Euler characteristic. Section 1.1 contains more detailed pseudocode for the method presented in Section 1.0.2. For dimension 2 we have two alternative methods for estimating Euler characteristic; although *these methods fare better in practice than inclusion-exclusion, they do not obviously generalize to higher dimensions.* Section 2 contains a "sweeping line", or Morse-theoretic, approach to estimating the Euler characteristic in $\mathbb{R}^2$. Section 3 outlines a third algorithm for the 2D case based on a Reeb graph construction. Our implementation and results are summarized in Section 5, and Section 6 concludes the paper with some remarks about further directions of research.

Acknowledgement: This research is supported in part by NSF/DARPA Grant #0310589. The authors are also grateful to Prof. Mufeed Mahd for MRI heart data.

1 The Euler Characteristic

We work in Euclidean affine space $\mathbb{R}^n$. A 0-cell is a point of $\mathbb{R}^n$. A set is called a (topological) k-cell if it is homeomorphic to an open Euclidean ball of dimension k. For example, relatively open convex sets in $\mathbb{R}^n$ are examples of cells.

Let $\mathcal{C}$ be a collection of topological cells in $\mathbb{R}^n$ such that the intersection of a finite number of elements of $\mathcal{C}$ can always be represented as a disjoint finite union of elements of $\mathcal{C}$.

Consider $\mathfrak{A}_{\mathcal{C}}$, the boolean algebra (i.e. a system closed with respect to finite unions and intersections) generated by $\mathcal{C}$. The boolean algebra generated by relatively open convex sets *OpenConv*, the boolean algebra of semi-algebraic sets *SemiAlg*, and the boolean algebra of sub-analytic sets *SubAn* will serve for us as main examples of $\mathfrak{A}_{\mathcal{C}}$. Sub-analytic sets include all semi-analytic and semi-algebraic sets. Sub-analytic sets provide enough flexibility for all conceivable practical situations in surface and solid modeling in $\mathbb{R}^n$.

Definition 1 *The Euler characteristic χ on $\mathfrak{A}_{\mathcal{C}}$ is a finitely-additive measure, which is defined on a k-cell $C \in \mathcal{C}$ as $(-1)^k$ for any $k \in \mathbb{Z}_{\geq 0}$.*

Of course, it is not clear from the above definition that χ exists. The proof of existence of χ for *OpenConv* can be found in [KR97], the proof of existence for *SemiAlg* can be found in [JB98], and that for *SubAn* in [Loj93].

Since $\mathfrak{A}_{\mathcal{C}}$ has a generating system of disjoint cells, then by defining χ on cells we define it on all of $\mathfrak{A}$ – the Euler characteristic of any $K \in \mathfrak{A}$, which is not a cell, can be computed by the inclusion-exclusion principle which holds for any finitely-additive measure:

$$\chi(K_1 \cup \cdots \cup K_m) \;=\; \sum_i \chi(K_i) \;-\; \sum_{i_1<i_2} \chi(K_{i_1} \cap K_{i_2}) \;+\; \cdots \;+\; (-1)^m \chi(K_1 \cap \cdots \cap K_m). \tag{1}$$

Our algorithms rely heavily on this **inclusion-exclusion** property of the Euler characteristic. Often the following proposition, which is just a restatement of the definition in our setup, is taken for the definition of Euler characteristic.

Proposition 2 *If $\mathcal{P}$ is a partition of $K \in \mathfrak{A}_{\mathcal{C}}$ that has f_0 points, f_1 1-dimensional cells, f_2 2-dimensional cell, and so on, up to f_n n-dimensional cells, then χ can be computed as*

$$\chi(K) = f_0(K) - f_1(K) + \cdots + (-1)^n f_n(K). \tag{2}$$

Then one has to show that this definition does not depend on the choice of the partition. One can define Euler characteristic by the above formula without introducing the algebra $\mathfrak{A}_{\mathcal{C}}$, via a triangulation (or, more generally, a cell-partition) of K, if it admits one. In this case f_i stands for the numbers of simplices (cells) of dimension i. If one chooses this route, the proof of invariance with respect to the choice of triangulation (cell-partition) can be found in e.g. [Mun84]).

In analogy to the Fubini theorem (or Cavalieri principle) for Euclidean volume, the Euler characteristic is also multiplicative over direct products: If $K \in \mathfrak{A}$ in $\mathbb{R}^k$, $L \in \mathfrak{A}$ in $\mathbb{R}^l$, then the Euler characteristic of $K \times L$ in $\mathbb{R}^{k+l}$ is given by

$$\chi(K \times L) = \chi(K)\chi(L). \tag{3}$$

In particular, if I denotes a closed interval in the real line, then $\chi(K \times I) = \chi(K)$. This formula can be derived using an inclusion-exclusion argument (see, for example, [KR97]).

Invariance with respect to homotopical equivalence (in the category of CW-complexes) is one of the most important properties of Euler characteristic (see [Mun84] for proofs).

Theorem 3 *Let $K_1, K_2 \in \mathfrak{A}_{\mathcal{C}}$ be finite CW-complexes with respect to some partitions into elements of $\mathcal{C}$. If they are homotopically equivalent, then $\chi(K_1) = \chi(K_2)$.*

A *region* of $\mathbb{R}^n$ is a pure n-dimensional (in the sense of Lebesgue) subset of $\mathbb{R}^n$. An n-dimensional *body* is a bounded connected closed region of $\mathbb{R}^n$.

In dimensions 2 and 3 the properties described above lead to especially simple and intuitive descriptions of the Euler characteristic. For the 2-dimensional case, suppose that $K \in \mathfrak{A}_{\mathcal{C}}$ is a region in $\mathbb{R}^2$ which is constructed by taking a closed region from $\mathfrak{A}_{\mathcal{C}}$ and creating a number of "holes" $H_i \in \mathfrak{A}_{\mathcal{C}}$ by subsets H_i, which are chosen to be open and contractible, in the interior of K. In this instance the original region has $\chi(K) = \#(\text{components of } K)$, while the creation of each "hole" involves the removal of a 2-cell, thereby decreasing χ by 1 for each hole (as in the formula (2)). This results in the following formula:

Proposition 4 *Let $K \in \mathfrak{A}_{\mathcal{C}}$ be a region in the plane. Then $\chi(K) = \#(\text{components of } K) - \#(\text{holes in } K)$.*

For the 3-dimensional case, suppose that $K \in \mathfrak{A}_{\mathcal{C}}$ is a region in $\mathbb{R}^3$ that is constructed by taking a finite union R of contractible regions from $\mathfrak{A}_{\mathcal{C}}$ and "drilling" a finite number of "tunnels" by removing solid open cylinders T_i—elements of $\mathfrak{A}_{\mathcal{C}}$—that pass entirely through the original region R, and creating a number of spherical "bubbles" B_j by hollowing out open contractible regions from the interior of R. Suppose the holes do not overlap. Each "tunnel" reduces χ by 1 (this follows from (1)), while each "bubble" created involves the removal of a 3-dimensional cell, thereby increasing the value of χ by 1 (as in the formula (2)). This results in the following formula for $\chi(K)$:

Proposition 5 *Let $K \in \mathfrak{A}_{\mathcal{C}}$ be a closed region of $\mathbb{R}^3$ constructed by removing the finite disjoint union of tunnels T_i and bubbles B_j from R. Then*

$$\chi(K) = \#(\textit{components of } K) - \#(\textit{tunnels in } K) + \#(\textit{bubbles in } K).$$

Keep in mind that Proposition 5 does not apply to *all* regions of $\mathbb{R}^3$, but only to regions with disjoint tunnels and bubbles of the particular types outlined above. For example, Proposition 5 would not apply to a ball-like region with a toroidal (doughnut-shaped) bubble – in this instance a more careful use of (1), (2), and (3) would be required. Indeed, a spherical region with a toroidal (doughnut-shaped) bubble is homotopy-equivalent to a hollow sphere with a hollow ring attached at one point, so that $\chi = \chi(\text{hollow sphere}) + \chi(\text{hollow ring}) - \chi(\text{point}) = \chi(\mathbb{S}^2) + \chi(\mathbb{S}^1) - \chi(\mathbb{R}^0) = 2 + 0 - 1 = 1$.

A method for estimating or computing the Euler characteristic in dimension $n-1$ leads in turn to a method for estimating or computing the $(n-1)$-dimensional *surface area* A_{n-1}. Let $\operatorname{lin} \mathbf{u}$ denote the line passing through the origin and parallel to the vector $\mathbf{u}$.

Theorem 6 (Cauchy-Crofton surface area formula) *Let $K \in \mathfrak{A}_{\mathcal{C}}$ be a body in $\mathbb{R}^n$. Then*

$$A_{n-1}(K) = \frac{1}{2} \int_{\mathbb{S}^{n-1}} \int_{\mathbf{u}^\perp} \chi(\, K \cap (\, \operatorname{aff}\{x, x+\mathbf{u}\}\,)\,)\; dx\; d\mathbf{u}.$$

The theorem and its many generalizations [KR97, San76] offer methods of computing surface area (or perimeter), mean width, projections, quermassintegrals, and related geometric functionals, by averaging the Euler characteristic of slices of a region by flats (or other convenient sets).

Based on this theorem, we designed an algorithm for estimation of surface area of a 3D body K with positive local feature size. The input is a point sampling $S \subset K$. The algorithm is using a subroutine Perimeter, which computes the perimeter of 2D body of arbitrary topology using the Cauchy-Crofton formula. $\chi(K \cap \operatorname{aff}\{x, x+\mathbf{u}\})$ is evaluated, approximately, by replacing $K \cap \operatorname{aff}\{x, x+\mathbf{u}\}$ with $S \cap \{z \;\big|\; 0 \le \mathbf{u}^\perp \cdot (z-x) \le \delta\}$ and using our algorithm for Euler characteristic in $\mathbb{R}$. Then $\int_{\mathbf{u}^\perp} \chi(K \cap \operatorname{aff}\{x, x+\mathbf{u}\})\; dx$ is replaced with a finite sum

$$\sum_i \overline{\chi}(\, S \cap \{\, z \in \mathbb{R}^2 \;\;|\;\; \mathbf{u}^\perp \cdot x_i \le \mathbf{u}^\perp \cdot z \le \mathbf{u}^\perp \cdot x_{i+1} \,\}\,) = \overline{\chi}_{\mathbf{u}},$$

where $x_{i+1} - x_i = \delta$ and the summation is over those i for which the intersection of the point sampling S and the strip $\{z \;\big|\; \mathbf{u}^\perp \cdot x_i \le z \cdot \mathbf{u}^\perp \le \mathbf{u}^\perp \cdot x_{i+1}\}$ is non-empty. $\int_{\mathbb{S}^{n-1}} \ldots d\mathbf{u}$ is replaced by $2\pi \sum_{\mathbf{u} \in \mathrm{Dir}} \overline{\chi}_{\mathbf{u}}$, where Dir is a finite set of unit vectors, which is either the vertex set of a regular convex polygon with $|\,\mathrm{Dir}\,|$ vertices, or a sampling from a uniform distribution on $\mathbb{S}^1$.

For direction of integration $\mathbf{d}$ the surface area of $K \subset \mathbb{R}^3$ is approximated by

$$(4) \qquad \sum_j \delta \textsc{Perimeter}[\ \mathrm{P}_{\mathbf{d}^\perp}(\ S \cap \{\ z \quad | \quad \mathbf{d}\cdot x_j \le z\cdot\mathbf{d} \le \mathbf{d}\cdot x_{j+1}\ \}\ )\ ] + \mathit{CorrectionTerm},$$

where $\mathbf{d} \in \mathbb{R}^3$, $\mathrm{P}_{\mathbf{d}^\perp}$ is the projection on $\mathbf{d}^\perp$, and $\mathbf{x}_j$'s ($j \in \mathbb{Z}$) form a partition of $\operatorname{lin}\mathbf{d}$ of size δ. $\mathit{CorrectionTerm}$ helps improve the precision by more accurate evaluation of the area contribution of those slabs of K that have significant value of

$$A_2(\ \mathrm{P}_{\mathbf{d}^\perp}\ (\partial K \cap \{\ z \in \mathbb{R}^3 \quad \big| \quad \mathbf{d}\cdot x_j \le z\cdot\mathbf{d} \le \mathbf{d}\cdot x_{j+1}\ \}\ )\ );$$

for example, $\sum_j \delta\textsc{Perimeter}[\ldots]$ ignores the contribution of two facets of the cube $[0,1]^3$, if $\mathbf{d} = \mathbf{e}_i$, where $\mathbf{e}_i$ is a vector of an orthogonal frame.

Finally, the estimates given by formula (4) for various directions $\mathbf{d}$ are averaged; the directions are drawn from the uniform distribution on the sphere $\mathbb{S}^2$.

1.0.1 Estimating χ on the Real Line

The first and fundamental case is that of the real line. Suppose a collection of sample points is given from an unknown set $K \subseteq \mathbb{R}$. It is assumed that K is a finite union of closed bounded intervals. The Euler characteristic of K is the number of disjoint closed intervals that comprise the set K.

To estimate this number, first choose a threshold value $\delta_1 > 0$. If x, y are two sample points and $|x - y| \le \delta_1$, then x and y are considered to lie in the same closed interval of K.

More generally, define an equivalence relation on sample points as follows: For sample points x, y we will say that $x \overset{\delta_1}{\sim} y$ if and only if there exists a sequence of sample points $x = x_0, x_1, \ldots, x_k = y$, such that $|x_{i+1} - x_i| \le \delta_1$ for each consecutive pair.

This equivalence relation defines a partition of sample points into a finite family of disjoint equivalence classes, corresponding to a finite number of closed intervals. The Euler characteristic $\chi(K)$ is then estimated to be the number of those closed intervals induced by this equivalence relation.

1.0.2 Estimating χ via Inclusion-Exclusion

Suppose a collection of sample points S is given from an unknown set $K \subseteq \mathbb{R}^n$, where $n \ge 2$. We also assume, without loss of generality, that all sample points have positive coordinates – translations do not change the Euler characteristic.

We will define an algorithm for estimating $\chi(K)$, assuming we already have an effective algorithm for estimating suitable classes of sets in the lower dimensional space $\mathbb{R}^{n-1}$, which depends on the vector of threshold parameters $\Delta = (\delta_1, \ldots, \delta_{n-1})$ and the vector of overlap parameters $\mathbf{p} = (p_2, \ldots, p_{n-1})$, which are introduced below.

Choose an orthogonal frame for $\mathbb{R}^n$. Denote the first coordinate axis of $\mathbb{R}^n$ by L and let $H = L^\perp$, the orthogonal complement of L. In other words, $L = \{(x_1, 0, \ldots, 0) \mid x_1 \in \mathbb{R}\}$ and $H = \{(0, x_2, \ldots, x_n) \mid x_i \in \mathbb{R}\}$. Evidently H is linearly isomorphic to $\mathbb{R}^{n-1}$.

Express the positive half of the axis $L \cong \mathbb{R}$ as a union of equally spaced overlapping intervals as follows: Choose real parameters $\delta > 0$ and $0 << p \le 1/2$. Let $I_0 = [0, \delta]$ and let

$$I_m = [m(1-p)\delta, m(1-p)\delta + \delta] \text{ for } m \in \mathbb{Z}_{\ge 0}$$

Each of the intervals has length δ. Moreover, the intersections $J_m = I_m \cap I_{m+1}$ of adjacent overlapping intervals are themselves closed intervals of length $p\delta$.

For any interval I_m of the axis L, let V_m denote the "vertical" slab of $\mathbb{R}^n$ parallel to the hyperplane H. That is,

$$V_m = I_m \times H = \{(x_1, x_2, \ldots, x_n) \mid x_1 \in I_m \text{ and } (x_2, \ldots, x_n) \in H\}.$$

Let $S(V_m)$ denote the set of sample points in this slab, that is, $S(V_m) = S \cap (I_m \times H)$. Project the points of $S(V_m)$ orthogonally onto H, and apply the algorithm for estimating the Euler characteristic in lower dimensions to the resulting cloud of points in H. Denote the resulting value by $f(I_m)$. The value $f(I_m)$ is a crude estimate of the Euler characteristic of $K \cap (I_m \times H)$, that is, a slice of K parallel to the hyperplane H. By examining how $f(I_m)$ varies from interval to interval, we will develop a more precise estimate of $\chi(K)$, exploiting the inclusion-exclusion and homotopy-invariance properties of the Euler characteristic.

To compute this estimate effectively, we must account for the (deliberate) fact that each adjacent pair of sample intervals overlap in an interval J_m of length $p\delta$. Recall from the inclusion-exclusion identity (1) that if $K, L \in \mathfrak{A}_{\mathcal{C}}$, then $\chi(K \cup L) = \chi(K) + \chi(L) - \chi(K \cap L)$. Iterating this inclusion-exclusion identity over the union $K = \bigcup_{m=0}^{\infty}(K \cap I_m)$ we find that

$$\chi(K) = \sum_{m \geq 0} \chi(K \cap I_m) - \sum_{m \geq 0} \chi(K \cap J_m) + \textit{vanishing terms}$$

Our choice of the intervals I_m guarantees that for $p \leq 1/2$ the intersections of 3+ intervals are either empty or cancel out. (The choice of p is left to the user of the software; in all of the reported tests we used $p = 1/2$.) Since K is bounded (and the given sample set of points in K is finite) the sums and unions above are *finite*. In other words, for some $M > 0$ we will have $K \cap I_m = \varnothing$ and $\chi(K \cap I_m) = 0$ for all $m > M$.

Without loss of generality assume that the sample point with the smallest x_1 has $0 < x_1 < (1-p)\delta$. Choose minimal M so that the coordinates of all sample points of K along the axis L lie inside the interval $[0, M(1-p)\delta + \delta]$. The value of $\chi(K)$ is then estimated to be

$$\overline{\chi}(K) = \sum_{m=0}^{M} [f(I_m) - f(J_m)]$$

The following applies not only to the inclusion-exclusion method, but to all of our algorithms. It is certainly possible for this estimate to be incorrect, especially if the threshold value δ is chosen too small or too large. Since the number of sample points is much larger than $\chi(K)$, choosing too small δ may result in a much grosser error than choosing too large δ. In order to improve the quality of the estimate, the entire procedure should be performed in many rotated frames, and the resulting values then averaged and rounded to the nearest integer (since the Euler characteristic $\chi(K)$ is *always an integer value*).

1.1 Inclusion-exclusion Algorithm for $\chi(K)$

We now describe in greater detail how we implement estimation of the Euler characteristic of a region in $\mathbb{R}^n$, given partial data in the form of point samples. For notational simplicity and to

reflect our actual implementation the pseudo-code below in Algorithm 1 refers to the 3-dimensional case. Although we implemented it for $n \leq 3$, our implementation can be adopted for the case when the dimension n is one of the input parameters. This is the only practical approach, that we found so far, for $n \geq 3$. Although the 2D algorithms in the subsequent sections give better speed and precision in practice, *they do not seem to generalize easily even to the 3D case.*

Let $\mathrm{MAX}(S, \mathbf{d})$ be a procedure that takes as input $S \subset \mathbb{R}^n$, and produces the element with the highest value of $\mathbf{d} \cdot \mathbf{x}$ among all $\mathbf{x} \in S$. $\mathrm{MIN}(S, \mathbf{d})$ is defined in the similar way. $\mathrm{Order}(S, \mathbf{d})$ is a procedure that takes $S \subset \mathbb{R}^d$ and produces a linked indexed (by elements of $\mathbb{Z}_{\geq 0}$) list, of length $|S|$, where indices are ordered by the value of $\mathbf{d} \cdot \mathbf{x}$; each node contains a point $\mathbf{x} \in \mathbb{R}^d$ and the value of $\mathbf{d} \cdot \mathbf{x}$, which is the key of the node. $\mathrm{Sublist}(List, l, r)$ takes an indexed $List$ of pairs $(\mathbf{x}, w)$, with $\mathbf{x} \in \mathbb{R}^d$ and $w \in \mathbb{R}$, and adds to each node $List[i]$, with $w \in [l, r]$, links to the nearest preceding and consecutive elements of $List$ with the same property $w \in [l, r]$. $T_{\mathbf{d}}$ stands for the transformation that projects $\mathbb{R}^3$ on the linear subspace $\mathbf{d}^{\perp}$ along $\mathbf{d}$.

X-3D-via-IE$(S, \Delta, \mathbf{p}, \#3Ddir, \#2Ddir; 2DMethod)$ is the recursive Inclusion-Exclusion algorithm for statistical estimation of χ for a bounded closed region $K \subset \mathbb{R}^3$ given as a point sampling $S \subset K$. Here $\Delta = (\delta_1, \delta_2, \delta_3)$ is the vector of threshold values for $n \in \{1, 2, 3\}$, $\#3Ddir$ is the number of directions of integration in $\mathbb{R}^3$, and $\#2Ddir$ is the number of directions for integration of projections of slabs on a hyperplane parallel to them. X-3D-via-IE calls a procedure X-2D-Random$(S_2, \Delta, \mathbf{p}, \#2Ddir; 2DMethod)$, which estimates χ of a 2D region by averaging the results of numeric Euler integration by method $2DMethod$ of the point cloud S_2 over $\#2Ddir$ random directions. $2DMethod$ is a parameter of X-2D-Random and represents user's choice for one of the three 2D algorithms that we designed. The universal inclusion-exclusion procedure, described in Section 1.0.2 is one of the possible choices for $2DMethod$. The other two possible choices, given in Sections 2, 3, can be used to validate our results, giving data for comparison against the inclusion-exclusion procedure. $\mathbf{p} = (p_3, p_2)$ is the vector of overlap parameters for 3D and 2D (if chosen) inclusion-exclusion procedures. Each of the 2D procedures make use of the *unique* algorithm for $\mathbb{R}$, which was described earlier.

2 A Morse-Theoretic Approach in the Plane

In this section we will outline a different approach to estimating the Euler characteristic of regions in the plane $\mathbb{R}^2$, using a "sweeping line" or "Morse theoretic" perspective. We have implemented this algorithm in $\mathbb{R}^2$. As for dimensions $n \geq 3$, it appears that the "sweeping hyperplane" approach will require a much more careful analysis of a great number of cases, depending on estimates of local curvature. Such computations appear to be much less reliable and more expensive which makes them impractical already for $n = 3$.

This alternative approach is based on the following interpretation of the Euler characteristic. Let $K \in \mathfrak{A}_{\mathcal{C}}$ be a bounded region in the plane. For any fixed $x \in \mathbb{R}$ let $\ell_x = \{(x, y) | y \in \mathbb{R}\}$, a vertical line in $\mathbb{R}^2$, and let $f_K(x) = \chi(K \cap \ell_x)$. Since $K \cap \ell_x$ is one-dimensional, $f_K(x)$ measures the number of closed line segments that form the linear section $K \cap \ell_x$ of K. For $x \in \mathbb{R}$ define $f_K(x+) = \lim_{\varepsilon \to 0^+} f_K(x + \varepsilon)$. It can be shown (see [KR97, p. 46]) that

$$\chi(K) = \sum_{x \in \mathbb{R}} [f_K(x) - f_K(x+)] \tag{5}$$

Remark: Although it might appear as though the expression in (5) represents an uncountable (and possibly undefined) sum, in fact this sum is always *finite*, since $f_K(x) - f_K(x+) \neq 0$ at only a

Algorithm 1 X-3D-via-IE($S, \Delta, \mathbf{p}, \#3Ddir; 2DMethod, \#2Ddir$)

$STACK3D \leftarrow \#3Ddir$ random directions in $\mathbb{R}^3$; $SumOverDirs \leftarrow 0$;
while $STACK3D \neq \varnothing$ **do**
  $\mathbf{d} \leftarrow \text{Pop}(STACK3D)$; $List \leftarrow \text{Order}(S, \mathbf{d})$;
  $L \leftarrow \text{MAX}(S, \mathbf{d}) - \text{MIN}(S, \mathbf{d})$; {length of body in direction $\mathbf{d}$ }
  $\#SlabsToGo \leftarrow \lceil \frac{\frac{L}{\delta_3} - p_3}{1 - p_3} \rceil$;
  $l \leftarrow \text{MIN}(S, \mathbf{d})$; $r \leftarrow l + \delta_3$;
  $DirEst \leftarrow 0$; $EstSlab \leftarrow 0$; $EstOverlap \leftarrow 0$;
  while $\#SlabsToGo > 0$ **do**
    $Slab \leftarrow \text{Sublist}(List, l, r)$;
    $EstForSlab \leftarrow \text{X-2D-Random}(T_{\mathbf{d}}(Slab), \Delta, \mathbf{p}, \#2Ddir; 2DMethod)$;
    $DirEst \leftarrow DirEst + EstForSlab - EstOverlap$;
    $Overlap \leftarrow \text{Sublist}(Slab, r - p_3\delta_3, r)$;
    $EstOverlap \leftarrow \text{X-2D-Random}(T_{\mathbf{d}}(Overlap), \Delta, \mathbf{p}, \#2Ddir; 2DMethod)$;
    $l \leftarrow r - p_3\delta_3$; $r \leftarrow l + \delta_3$;
    $\#SlabsToGo \leftarrow \#SlabsToGo - 1$;
  end while
  $SumOverDirs \leftarrow SumOverDirs + DirEst$;
end while
X-3D-via-IE $\leftarrow \frac{SumOverDirs}{\#3Ddir}$;

finite number of critical values of x. For example, suppose K is a convex polygon and lies over the closed interval $[a, b]$ of the x-axis. In this instance, $f_K(x+) = f_K(x) = 0$ if $x < a$ or $x > b$, while $f_K(x+) = f_K(x) = 1$ if $a \leq x < b$. Only $x = b$ do we have $f_K(x) = 1$ and $f_K(x+) = 0$, so that

$$\sum_{x \in \mathbb{R}} [f_K(x) - f_K(x+)] = f_K(b) - f_K(b+) = 1 - 0 = 1.$$

The identity (5) suggests that the Euler characteristic of a region K can be computed by sweeping a line across the region and keeping careful account of when and how the associated cross-section of K changes topologically; that is, how the number of connected intervals of the cross-section changes as the line sweeps by. We will now offer a discrete implementation of this technique.

2.1 Sweeping Line Algorithm for Point Clouds in the Plane

Suppose a collection C of sample points is given from an unknown set $K \subseteq \mathbb{R}^2$. Once again we assume that the actual set K is a finite union of bounded planar regions. Choose a randomly oriented orthogonal frame $[\mathbf{o}; \mathbf{e}_1, \mathbf{e}_2]$ so that all sample points have positive coordinates, and let (x, y) denote a point $\mathbf{o} + x\mathbf{e}_1 + y\mathbf{e}_2$.

We will refer to the *x-axis* of this frame as the horizontal axis, and the *y-axis* as the vertical axis. First, the vector of thresholds $(\delta_1, \delta_2, \tau, T)$, where $\tau > \delta_1, \delta_2$ and $T > \tau$, is chosen by the user of the algorithm. Parameters δ_1 and δ_2 have the role similar to that of δ_1 in Section 1.0.2. T and τ are sensitivity parameters controlling matching of fragments in adjacent vertical strips; their exact meaning is given by the pseudocode below in Algorithm 2.

Let us now cover the positive ray of the x-axis with closed intervals I_j, where $j = 0, 1, 2, \ldots$, each of length δ_2, with non-overlapping interiors. For each interval I_j of the x-axis, let $V_j = I_j \times \mathbb{R}$

denote the vertical strip of $\mathbb{R}^2$ that projects onto I_j, and let C_j denote the set of sample points in this strip, that is, $C_j = C \cap V_j$. Project the points of C_j horizontally onto the y-axis, and apply the algorithm for estimating the Euler characteristic in the real line with threshold value of δ_1 (see above) to the resulting cloud of points on the y-axis. Denote the resulting value by $f(j)$. The value $f(j)$ is a crude estimate of the Euler characteristic of $K \cap V_j$. By examining how $f(j)$ varies from interval to interval, we can develop a more precise estimate of $\chi(K)$, in the spirit of (5). We denote by $\operatorname{int} I$, where $I \subset \mathbb{R}^n$, the interior of set I. To each pair (C_{j-1}, C_j) we assign integer $g(j)$, which is a statistical estimate of non-zero terms $[f_K(x) - f_K(x+)]$ in formula (5) for $x \in \operatorname{int} I_{j-1} \cup \operatorname{int} I_j \cup (I_{j-1} \cap I_j)$. Note that $g(j)$ is zero if both C_{j-1} and C_j are empty; however, g_j may be zero even when C_{j-1} and C_j are not empty.

Denote by $\bar{\chi}(K)$ the estimate of $\chi(K)$ obtained by numeric Euler integration in the direction of the x-axis. Then

$$\bar{\chi}(K) = \sum_{\{j \in \mathbb{N} \mid |f(j-1)|, |f(j)| > 0\}} g(j).$$

Let us now explain how we compute $g(j)$. Each interval I_j has been assigned an integer value $f(j)$. Suppose that I_{j-1} and I_j are two consecutive intervals along the x-axis. It is convenient to think of j as of a value of discrete time $t \in \mathbb{N}$. Then any change in the topology of $K \cap V_t$, as t changes from $j-1$ to j is called an *event*. Let us call the connected components of $K \cap V_j$ *fragments.*

For each $j \in \mathbb{Z}_{\geq 0}$ such that $V_j \cap S \neq \varnothing$, we guess the partition of the actual subset $K \cap V_j$ into fragments by grouping points according to their y-coordinate and applying the 1-dimensional algorithm to the resulting cloud of points on the y-axis. Thus, we can think that we deal with two families of real intervals on the y-axis, one coming from C_{j-1} and the other from C_j. We call the first family *intervals on the left* and the second *intervals on the right*.

Recall that the strips are presumed to be thin when compared with the minimal feature size of the body K, and that the direction of the strip sides is random. It can then be argued that in such a case, generically, only one "event" can occur during the transition from V_{j-1} to V_j. By an *event* we mean one of the following: *birth* of a new fragment in V_j (a connected component of $K \cap V_j$ that does not abut any connected component of $K \cap V_{j-1}$), the *death* of a fragment of V_{j-1} (a connected component of $K \cap V_{j-1}$ that does not abut any connected component of $K \cap V_j$), the *split* of a fragment of V_{j-1} into two fragments in V_j (two connected components of $K \cap V_j$ abut one connected component of $K \cap V_{j-1}$), or the *merge* of two neighbor fragments of V_{j-1} into one fragment in V_j (two connected components of $K \cap V_{j-1}$ abut one connected component of $K \cap V_j$). If the direction of integration is reversed, all birth events becomes death events and all death events become birth events; the same holds for merges and splits.

If, indeed, at most one event occurs, then the number of fragments in V_j differs from the number of fragments in V_j by at most one – this is the case of *generic transition*. Otherwise we say that transition from V_{j-1} to V_j involves multiple events – such a transition will be called non-generic.

Transitions can be handled in two different ways. Under the first, *lazy,* approach we first compute $f(j) - f(j-1)$ and if it is zero, we just assume that no event occurred; if $|f(j) - f(j-1)| \geq 1$, we turn to the *strict* approach, which is described below. The lazy approach allows us to save a lot of computing time, but, potentially, introduces errors. Our experiments show that averaging over a number of random directions takes care of these errors. The lazy approach is justified under the

assumptions that all transitions are generic.

Under the strict approach we match the fragments in V_{j-1} and V_j and determine all events in the transition. For success of numerical Euler integration, no matter what method is used, the number of strips should be large compared to the number of critical points of function $\mathbf{d} \cdot \mathbf{x}$, restricted to K, where $\mathbf{d}$ is a vector of direction of integration. In such a case for almost all values of j we have $f(j) - f(j-1) = 0$ and that is why the lazy approach is much faster. So, we have a classic time-accuracy tradeoff.

2.1.1 Transition Analysis

Let $Left$ and $Right$ be two adjacent vertical strips. We say that fragment F in $Left$ matches fragments $F_1, F_2, \ldots$ in $Right$, or, vice versa – $F_1, F_2, \ldots$ match F, if the projections of $F_1, F_2, \ldots$ on the y-axis overlap with the projection of F on this axis.

The estimated value of $\chi(K)$ is given by $\overline{\chi}(K) = \sum_{j=1}^{\infty} g(j)$. Since $g(j) \neq 0$ for only finitely many values of j, the sum is always well-defined.

Once again this estimate may be incorrect, especially if the threshold values are chosen too large or too small. In order to improve the quality of the estimate, the entire procedure might be performed in many randomly rotated frames, and the resulting values then averaged and rounded to the nearest integer (since the Euler characteristic $\chi(K)$ is *always* an integer value).

Algorithm 2 below provides details for computing $g(j)$. To simplify the notation we do not use index j in the pseudocode for MATCHING; i.e. instead of $g(j)$ we just write g. Suppose the points in $S \cap Left$ have been clustered into groups using the 1-dimensional Euler integration on the set of their y-coordinates and that this clustering has been recorded as a stack of intervals on the y-axis: $S_L = ([l_1, u_1] < \ldots < [l_k, \leq u_k]$. Here $[l_1, u_1]$ is the top of the stack and we write $[a, b] > [c, d]$ if and only if $a > d$.

Application of 1-dimensional Euler integration to new strip

First we apply Euler integration to the set of y-coordinates of points of $Right \cap S$. Record the end points of the resulting intervals as a stack $S_R = ([\lambda_1, \upsilon_1] < \ldots < [\lambda_k, \upsilon_k])$, where $[\lambda_1, \upsilon_1]$ is on the top of the stack.

Matching fragments in adjacent strips

Procedure MATCHING estimates how the Euler characteristic changes as time t changes from $j-1$ to j. The input to this procedure is two stacks S_L and S_R, of even lengths $2k$ and $2m$, of positive real numbers and tolerances τ and T. The output is the value of g. For an interval I attributes lower[I] and upper[I] contain the lower and upper ends of I, respectively. A procedure Top just accesses the top of the stack, without removing it, while Pop accesses the top and removes it from the stack. We say that an interval has been *called* if it has been passed by either Pop or Top at least once.

Procedure MATCHING makes use of a predicate OVERLAP(I, J, τ, T), which determines if there is sufficient statistical evidence that two fragments F_I and F_J in adjacent vertical strips, whose projections on the y-axis are I and J, represent pieces of the figure K that are connected inside these strips. If τ is set to 0 then OVERLAP$(I, J, 0, T) \equiv (I \cap J)$. Otherwise, it attempts to determine

Algorithm 2 MATCHING(S_L, S_R, τ, T)

$g \leftarrow 0$ {initialize the increment with 0}
LastRight $\leftarrow \varnothing$ {*LastRight* is a variable that holds the uppermost *called* interval on the right that overlaps with the last processed interval on the left and whose upper end is above the upper end of the latter; otherwise *LastRight* is $\varnothing$}
$l \leftarrow -\infty$ {convenient initialization}
if S_R is non-empty **then**
  while S_L is non-empty **do**
    $I \leftarrow \text{Pop}(S_L)$ {I is the lowest interval on the left, not called before}
    if *LastRight* $\neq \varnothing$ **then**
      if OVERLAP$(I, \textit{LastRight}, \tau, T)$ **then**
        $g \leftarrow g - 1$
      end if
      if upper(*LastRight*) $<$ upper(I) **then**
        {current *LastRight* cannot overlap with next interval on the left}
        LastRight $\leftarrow \varnothing$
      end if
    end if
    while $l <$ upper(I) **do**
      {while the lower end of next interval on the right is below the upper end of I}
      $J \leftarrow \text{Pop}(S_R)$
      {J is the lowest unprocessed interval on the right}
      if OVERLAP$(I, J, \tau, T) = 0$ **then**
        {a new branch has been born on the right, lying entirely below I, i.e., $I > J$}
        $g \leftarrow g + 1$ {J can't overlap with any element of S_L smaller than I, for *LastRight* $= \varnothing$ }
      else if upper$(I) <$ upper(J) **then**
        LastRight $\leftarrow J$
      end if
      if S_R is not empty **then**
        $l \leftarrow \text{lower}(\text{Top}(S_R))$ {l is assigned the lower end of the top element of S_R}
      else
        $l \leftarrow +\infty$
      end if
    end while
  end while
end if
{Enumerating remaining intervals on the right that do not overlap with any intervals on the left:}
while S_R is not empty **do**
  $\text{Pop}(S_R)$
  $g \leftarrow g + 1$
end while
return g

if the overlap is significant enough to infer connectivity. The pseudocode for OVERLAP is given after that of MATCHING.

The following is a boolean predicate whose input is two closed intervals in $\mathbb{R}$ and two real numbers τ and T, such that $0 << \tau < T$.

Algorithm 3 OVERLAP$([l,u],[\lambda,\upsilon],\tau,T)$

if $\upsilon \leq l$ **OR** $u \leq \lambda$ **then**
  $r \leftarrow 0$
else if $\big|\,[l,u] \cap [\lambda,\upsilon]\,\big| < \tau$ **then**
  {overlap is tiny on the absolute scale}
  if $\min(u-l, \upsilon-\lambda) > T$ **then**
    {intervals are big as compared to T}
    $r \leftarrow 0$
  else if $\frac{\big|\,[l,u]\cap[\lambda,\upsilon]\,\big|}{\min(u-l,\upsilon-\lambda)} < \frac{\tau}{T}$ **then**
    {overlap is small, even relative to T}
    $r \leftarrow 0$
  else
    $r \leftarrow 1$
  end if
else
  $r \leftarrow 1$
end if
return r

3 Reeb Graph Approach to the Case of $\mathbb{R}^2$

For a point sampling from a 2-D body K we have implemented an algorithm based on the idea of construction of a Reeb graph. Let $K \subset \mathbb{R}^2$, let $h_{\mathbf{d}} : K \longrightarrow \mathbb{R}$ be the height function $h_{\mathbf{d}}(\mathbf{x}) = \mathbf{d} \cdot \mathbf{x}$ for the direction $\mathbf{d} \in \mathbb{R}^2$, and let $G(h_{\mathbf{d}})$ be the graph of $h_{\mathbf{d}}$ in $K \times \mathbb{R}$. The Reeb [Ree46] complex of K for $h_{\mathbf{d}}$, which we denote by $\mathrm{Reeb}_{\mathbf{d}}(K)$, is $G(h_{\mathbf{d}})$ factored by the following equivalence relation: $(\mathbf{x}, h_{\mathbf{d}}(\mathbf{x})) \sim (\mathbf{x}_1, h_{\mathbf{d}}(\mathbf{x}_1))$ iff $h_{\mathbf{d}}(\mathbf{x}) = h_{\mathbf{d}}(\mathbf{x}_1)$ and both $\mathbf{x}$ and $\mathbf{x}_1$ lie in the same connected component of $h_{\mathbf{d}}^{-1}(h_{\mathbf{d}}(\mathbf{x}))$. It is known [Ree46] that the Reeb complex is homotopically equivalent to K (in fact, any continuous function can be used instead of $h_{\mathbf{d}}$). A finite simple graph $R(V, E)$ is called a Reeb graph of K for the function $h_{\mathbf{d}}$, if $R(V, E)$ is homeomorphic to $\mathrm{Reeb}_{\mathbf{d}}(K)$. Reeb graphs have found earlier use in computational topology (see, for example [EHMP04, WHDS04]), but not in meshless setups.

Let $S \subset K$ be a sampling of K, and let $\mathbf{d} \in \mathbb{R}^2$ be the direction of Euler integration; as usual, $\mathbf{d}$ is assumed generic, i.e. no two points of S have the same dot product with $\mathbf{d}$. The algorithm X-2D-via-Reeb$(S, \delta, \mathbf{d})$ produces an estimate for $\chi(K)$; it also gives estimates for the Reeb graph and boundary curves of K. As in the inclusion-exclusion and Morse approaches, we consider strips $V_j = I_j \times \langle \mathbf{d}^{\perp} \rangle_{\mathbb{R}}$ of width δ, where $\mathbf{d}^{\perp}$ is a unit vector perpendicular to $\mathbf{d}$. The idea of the algorithm is to replace $S \cap V_j$ with the disjoint union of boxes $\coprod_{l=1}^{|B_j|} B_{jl}$, where

$$B_{jl} = I_j \times H_l = \{\, \mathbf{x} \;\big|\; \mathbf{x} \cdot \mathbf{d} \in I_j,\ \ \mathbf{x} \cdot \mathbf{d}^{\perp} \in H_l \,\},$$

$H_l \subset \langle \mathbf{d}^{\perp} \rangle$, and B_j is the set of boxes in strip V_j.

Partition $V_j \cap S$ into $|B_j|$ equivalence classes C_{jl}: $\mathbf{u} \sim \mathbf{v}$ iff $\mathbf{d}^{\perp} \cdot \mathbf{u} \overset{\delta}{\sim} \mathbf{v} \cdot \mathbf{d}^{\perp}$. Then $B_{jl} = I_j \times \{\mathbf{x} \;\big|\; L_l \leq \mathbf{x} \cdot \mathbf{d}^{\perp} \leq H_l\}$, where $L_l = \min\{\mathbf{p} \cdot \mathbf{d}^{\perp} \,\big|\, \mathbf{p} \in C_{jl}\}$ and $H_l = \max\{\mathbf{x} \cdot \mathbf{d}^{\perp} \;\big|\; \mathbf{x} \in C_{jl}\}$. As the other algorithms, X-2D-via-Reeb$(S, \delta, \mathbf{d})$ proceeds by analyzing the transition from S_j to S_{j+1}. Denote by $\widetilde{R}$ the current state of the estimate $\overline{R}$. At first $\widetilde{R}$ consists of a finite set of disconnected edges, which corresponds to the intersection of S with the first non-empty strip. The number of edges is equal to the number of equivalence classes $|B_1|$. Without loss of generality, let us assume that $\mathbf{d}$ and $\mathbf{d}^{\perp}$ are collinear and co-oriented with the x- and y-axes. We say that there is a *match* between equivalence classes C_{jk} and $C_{j+1\,m}$, lying in two consecutive strips (and that box B_{jk} *matches* $B_{j+1\,m}$), if the right side of box B_{jk} has a non-empty intersection with the left side of box $B_{j+1\,m}$.

Since any particular Reeb graph of K is just a triangulation of $\mathrm{Reeb}_{\mathbf{d}}(K)$ and vertices of degree 2 in a Reeb graph do not carry any topological information, we will maintain $\widetilde{R}$ so that it does not contain such vertices. The graph $\widetilde{R}$ is represented by its adjacency-list; a vertex-node VN contains the name of its connected components of $\widetilde{R}$ and pointers to boxes from the last processed strip that have been assigned to VN.

Suppose we have just processed strip S_j. For each box B in S_j procedure Vertex(B) determines the vertex of $\widetilde{R}$ that corresponds to this box. Component(B) gives the name of the connected component of $\widetilde{R}$ that contains Vertex(B). The transition to S_{j+1} has three steps as follows:

(1) adding new edges to $\widetilde{R}$, disjoint from everything else—these edges represent the boxes in S_{j+1}; each of these edges defines a new connected component. For each new box $B \subset S_j$ let LV(B) and RV(B) be pointers to the vertices of a new edge of $\widetilde{R}$.

(2) matching equivalence classes of strip S_j and those of strip S_{j+1}. In both S_j and S_{j+1} the boxes are numbered in the decreasing order of value of x_2 and put into $STACK$, box #1 first. Next(B) gives the box with the next number or $\varnothing$, if B is the lowest box in the strip.

This step of X-2D-via-Reeb$(S, \delta, \mathbf{d})$ is given in the following pseudocode:

$BoxOnRight \leftarrow B_{j+11}$
while $STACK \neq \varnothing$ **do**
  $B \leftarrow$ Get$(STACK)$
  while $B \cap BoxOnRight \neq \varnothing$ and Next$(B) \neq \varnothing$ **do**
    Identify LV$(BoxOnRight)$ with Vertex(B)
    Component(RV$(BoxOnRight)$) $\leftarrow$ Component(B)
    $LowestBox \leftarrow BoxOnRight$
  end while
  $BoxOnRight \leftarrow LowestBox$
  $B \leftarrow$ Next(B)
end while

Let $MatchGraph$ be the graph of matches between boxes of S_j and S_{j+1}. The complexity of this procedure is obviously $O(|S \cap S_j| + |S \cap S_{j+1}|) + O(|MatchGraph|)$. The number of vertices of match graph is $|S \cap S_j| + |S \cap S_{j+1}|$. Since this graph is planar, its number of edges is linear in $|S \cap S_j| + |S \cap S_{j+1}|$.

(3) getting rid of vertices of degree 2 and updating $\widetilde{R}$.

After the algorithm has terminated we can count the number of connected components of $\overline{R}$ and compute $\overline{\chi} = \chi\overline{R}$.

4 Comparative Complexity

To simplify the presentation, let us presume that the integration is always being done in one direction, i.e. $\#dir2D = \#dir3D = 1$ in all instances. Suppose any arithmetic operation requires at most one unit of time. All presented algorithms require ordering the input. In all the algorithms the ordering happens to be the most time-consuming procedure from the viewpoint of worst-case asymptotic time complexity.

Case of $\mathbb{R}$. The estimation of $\chi(K)$, for $K \subset \mathbb{R}$, from a sampling of N points requires $\Omega(N \log_2 N)$ time and $\Omega(N)$ memory.

Case of $\mathbb{R}^2$. In all 2D methods the estimation of $\chi(S)$ for an individual strip S, given a point sampling of size s, requires $\Omega(s \log_2 s)$ time and $O(s)$ memory. Recording positions of groups of points (or boxes) along $\mathbf{d}^\perp$ takes time linear in s. The advantage of the inclusion-exclusion approach is that strips need not to be matched with the neighboring strip. However, the inclusion-exclusion method processes more strips than the other two methods, because of overlaps. If the length of the body in the direction $\mathbf{d}$ is $M\delta$ and the overlap between any two consecutive δ-segments is $p\delta/2$, with $p \leq 1$, then the inclusion-exclusion algorithm has to do with extra $2pM - 1$ strips of width δ and $M(2p + 1) - 2$ strips of width $p\delta/2$. E.g., for $p = 2/3$, used in the implementation, the number of strips processed by the inclusion-exclusion procedure is roughly 14/3 of the number of strips processed by either alternative; for $p = 1$, used in the pseudocode in 1, the number of strips to process is 6 times that for an alternative.

Set $\#CP_{\mathbf{d}} = |\{\tau \in \mathbb{R} \mid K \cap \{\mathbf{d} \cdot \mathbf{x} - t\}$ changes topology at $t = \tau\}|$, and let $\#CC_{\mathbf{d}}$ be the maximum number of connected components of $K \cap \{\mathbf{d}^\perp \cdot \mathbf{x} - t\}$ over $t \in \mathbb{R}$.

Since the 2D "sweeping line" approach presumes that at most one change can occur at the boundary of two strips, the transition analysis described in Section 2.1.1 is applied at most $\#CP_{\mathbf{d}}$ times. Each such analysis requires at most $const\#CC_{\mathbf{d}}$ operations. Thus, the cost of transition analysis is $O(\#CP_{\mathbf{d}}\#CC_{\mathbf{d}})$, which is indeed $O(N)$, where N is the number of sample points.

In the Reeb graph approach the transition analysis is done for all vertical lines that bound strips. If the number of strips is M, it takes $O(M\#CC_{\mathbf{d}})$, which is also $O(N)$.

Thus, all three 2D algorithms have worst case complexity of $\Omega(N \log_2 N)$.

Case of $\mathbb{R}^3$. The analysis of 3D inclusion exclusion algorithm is almost identical to that of the 2D inclusion-exclusion algorithm. The worst-case complexity is $\Omega(N \log_2 N)$.

In each of the algorithms for $\mathbb{R}^d$, for $d > 1$, certain generic position assumptions are made. For example, the assumption (1) *that no topological change happens on vertical $(d-1)$-planes separating slabs (strip for 2D),* is used by the inclusion-exclusion and Reeb graph procedures. The assumption (2) *that more than two events cannot happen in one slab* is used by the sweeping-line approach. They both can be violated in practice. The inclusion-exclusion procedure has no way of detecting such violations and thereby relies on averaging (which happens to be rather effective). In the case of (1) we can either shift a separator forward or backwards along $\mathbf{d}$, or perturb the positions of points that happened to lie on a separator. While the first approach, implemented in the Reeb graph algorithm, is less statistically biased, the second takes less time and programming efforts, since the shift of a separator means recalculation of $\overline{\chi}$ for at least two slabs. In the case of (2) we can either make slabs thinner, or choose a new direction of integration $\mathbf{d}$ and start anew, and that is what we have done in our implementation.

5 Experimental Results

Our implementation is in C++, using the C++ Standard Template Library. The code was compiled with Microsoft Visual C++ (with default optimization) on Windows PCs and g++ 3.2 on Linux i686 machines. Although the descriptions of our algorithms use real arithmetic, in practice, the coordinates of the sample points are either rational or integer. In our experiments the input points had either integral or rational coordinates (6 decimal digits) and we used double precision floats for evaluation of quadratic polynomials. All our algorithms are well-suited for floating point calculations, for the highest degree of evaluated predicates is 2.

5.1 Choice of Parameters

The user chooses the values of parameters, such as δ_1, etc. in each case. In our tests we used the same value of δ for all dimensions, i.e. $\delta = \delta_1 = \delta_2 = \delta_3$. Choosing δ too large will cause the algorithm to perceive the data as large simply-connected regions, whereas choosing δ too small will lead to an excessively granulated interpretation of the discrete partial data.

The user of the algorithms can choose δ_i on the basis of the expectations of the distance to the k-th nearest neighbor. For example, it is rather obvious that δ_3 should be significantly larger than $E_{N,1}$. Our experiments showed that for the uniform distribution δ_3 should be chosen to be at least 10 times $E_{N,1}$.

Let φ be any continuous well-behaved (e.g. uniform and Poisson are well-behaved) probability density φ on a compact body P, satisfying certain boundary conditions, and let ρ be any positive

number less than 1. Evans, A. Jones, and Schmidt showed that if N points are selected independently according to φ, then the expectation of the distance to the k-th nearest neighbor $E_{N,k}$ is given by

$$V_n^{-1/n}\frac{\Gamma(k+1/n)}{\Gamma(k)N^{1/n}}\int_P \varphi(\mathbf{x})^{1-1/n}\mathbf{dx} + O(\frac{1}{N^{2/n-\rho}}), \tag{6}$$

where V_n is the volume of the unit ball in $\mathbb{R}^n$ (see [EJS02]). It is sufficient if the following requirements are met: 1) φ is continuous, 2) partial derivatives of φ exist and are bounded in absolute value, 3) $\varphi > 0$. For the precise conditions on the body P see [EJS02]; we only note that all compact convex bodies and all bodies considered in our experiments satisfy them.

If P is a cube and the distances are measured under periodic boundary conditions, the second term can be improved to $O(\frac{1}{N^{1+1/n}})$ (see [PM98]). For data coming from applications $E_{N,k}$ should be estimated by taking the median of the distance to the k-th nearest neighbor. In practice, the median estimate agrees with the theoretical estimate much better than the mean value, since the mean value may be significantly affected by the boundary effect. All such statistics can be computed in $O(k\ N\ \log N)$ time.

Denote by $\mathcal{NN}_{\mathbf{x}}(\partial K)$ the set of all points of ∂K that are at the distance from $\mathbf{x}$ equal to the distance between $\mathbf{x}$ and ∂K. The *median axis* of K is defined as

$$\{\mathbf{x} \in \mathbb{R}^n \quad | \quad \operatorname{card}\mathcal{NN}_{\mathbf{x}}(\partial K) \geq 2\},$$

where card stands for cardinality. The local feature size of K, denoted by $\mathfrak{f}(K)$, is defined as the distance between the median axis of K and ∂K. The upper bound on admissible values of δ_i depends on the minimal feature size of K, the body under investigation. It is obvious from elementary geometric considerations that δ_3 should be less than $2\mathfrak{f}(K)$.

We are currently working on rigorous theorems for upper and lower bounds on admissible values of δ_i. These results will be published in a separate paper(s).

5.2 Test Results

We tested our algorithms on two types of data: uniform and $\mathbb{Z}^n$. The 3-dimensional data sets have been constructed by simulation[1], while the 2-dimensional test data was taken from MRI tomography. The analysis of results appears in Section 5.3.

All tables have the δ-column, the algorithm columns, corresponding to the choices of 2-dimensional algorithm, and the column(s) specifying the number of directions on the plane and in the space with which the algorithm was run. The algorithm columns contain time (in seconds) and Euler characteristic estimate for each run. In the tables below we denote by E the estimate given by the first term of formula (6) for $k = 1$. As in Section 5.1, $\mathfrak{f}(K)$ denotes the local feature size.

5.2.1 2D MRI Heart Data

Test results for the heart data are presented in Table 1. Our original data was a grey-scale image of cross-section of human heart (see left image on Figure 1). The input data for the algorithm was

[1]We were unable to obtain high-quality 3D MRI data corresponding to non-trivial topological shapes.

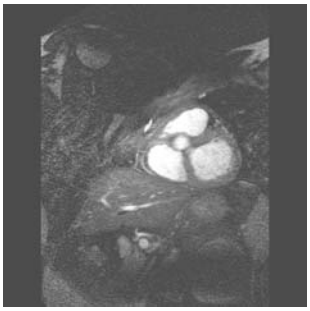
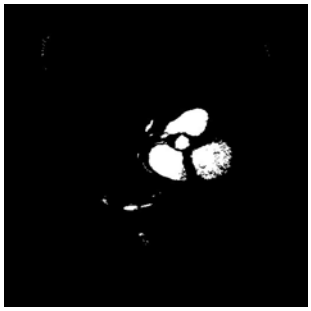

Figure 1: Human heart MRI cross-sectional data. Left: original grey-scale image. Right: black-and-white result of thresholding.

obtained from an .img file, for a threshold of 2,300,000 (see the right image on Figure 1). The range of intensities in the .img file was from 0 to 5,059,081. Each intensity corresponds to a point on a square fragment of $\mathbb{Z}^2$. The whole fragment is 51.09×51.09 and the density of sampling is about 100. The heart alone approximately fits in a square with side length of 17.5.

Column one holds the strip width $\delta = \delta_1 = \delta_2 = \delta_3$. The overlap parameters p_2 and p_3 for the inclusion-exclusion routines have been set to 1/2. The second column specifies the number of random directions, which have been drawn from a uniform distribution on the circle. Next, for each of the methods tested, subsequent pairs of columns give the values returned by 6 runs of each algorithm for the estimated Euler characteristic, given each particular choice of δ, along with the average running time for these trials in seconds. Varying choice of δ changes the ability of the algorithm to recognize the smaller white spots in Figure 1, accounting for the variation in the estimates $\overline{\chi}$.

Heart slice. Number of points before thresholding: 253632; image size: 51.09×51.09**;**							
minimal xy**-square containing heart:** 17.5×17.5**.**							
		Incl/excl		Reeb graph		Morse	
δ	#dir 2D	$\overline{\chi}$	Time	$\overline{\chi}$	Time	$\overline{\chi}$	Time
0.7	10	-4,-4,-4,-4,-4,-4	62	-4,-4,-4,-4,-4,-4	33	-4,-4,-4,-4,-4,-4	15
0.82	10	-4,-4,-4,-4,-4,-4	53	-4,-4,-3,-4,-3,-4	26	-4,-5,-3,-4,-5,-3	13
0.835	10	-5,-4,-4,-5,-5,-5	51	-3,-3,-3,-3,-3,-3	26	-2,-3,-2,-3,-2,-2	14
0.9	10	-2,-3,-3,-3,-3,-3	40	-3,-2,-4,-4,-4,-4	26	-3,-2,-2,-2,-2,-2	12

Table 1. Sampling: $\mathbb{Z}^2$; density ≈ 100; $E \approx 0.1$.
Computer: 32 bit i686, 2GB RAM, 2.8 GHz Xeon, Red Hat 8.0. g++: no optimization.
Parameters for Morse: $\tau = 0$, $T = \infty$, approach=lazy.
Time: average over 6 runs.

5.2.2 Cube with Three Tunnels

For the next test we took a cube of linear size 5 and carved in it three mutually perpendicular tunnels, each symmetric with respect to an axis passing through the centers of opposite faces. Each tunnel is formed by subtracting from the cube the union of 6 congruent balls $B_{\mathbf{c}}(1)$ of radius 1, with centers $\mathbf{c}$ lying on the corresponding axis. The cube is $[0,5] \times [0,5] \times [0,5]$. The centers of the

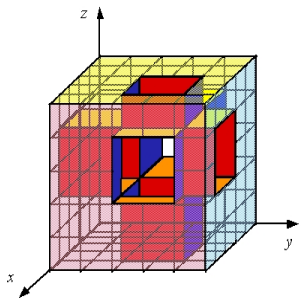

Figure 2: Cube with three crossing tunnels; $\chi = -4$.

balls all have integer coordinates varying from 0 to 5. More formally,

$$K = [0,5]^3 - \bigcup_{g \in \mathcal{S}_3} \bigcup_{n \in \{0,\ldots,5\}} B_{g(3,3,n)}(1)$$

where $\mathcal{S}_3$ stands for the symmetric group on 3 elements. Figure 2 shows a body homeomorphic to K.

True $\chi = -4$. 5x5x5 cube with 3 tunnels of diameter 1 meeting in the middle								
			Incl/excl		Reeb graph		Morse	
δ	#dir 2D	#dir 3D	$\overline{\chi}$	Time(s)	$\overline{\chi}$	Time(s)	$\overline{\chi}$	Time(s)
0.4	1	1	-4,-4,-5	1,2,1	-5,-4,-4	< 1,< 1,< 1	-4,-4,-4	< 1*
0.6	1	1	-3,-3,-4,-3	1,1,1,1	-4,-6,-4,-6	< 1*	-4,-5,-4,-4	< 1*
0.8	1	1	-4,-4,-4,-4	1,1,1,1	-5,-5,-4,-5	< 1*	-4,-3,-4,-4	< 1*
1.2	1	1	-3,-3	1,1	1,1	< 0.5 ,1	1,1	< 0.5,< 1

Table 2. Sampling: uniform; $N = 100000$, density=800, $E \approx 0.012856$.
Computer: 32 bit PC, 1.8 GHz Pentium 4, 256 RAM, XP Home.
Parameters for Morse: $\tau = 0$, $T = \infty$, approach=lazy.
*:for all four trials

True $\chi = -4$. 5x5x5 cube with 3 tunnels meeting in the middle								
			Incl-excl		Reeb graph		Morse	
δ	#dir 2D	#dir 3D	$\overline{\chi}$	Time(s)	$\overline{\chi}$	Time(s)	$\overline{\chi}$	Time(s)
0.5	5	10	-4,-4,-4	68,69,77	-4,-4,-4,	46,47,49	-4,-4,-3	26,27,27
0.6	5	10	-4,-4,-4	60,59,66	-4,-4,-4	41,42,44	-4,-4,-4	23,24,22
1	5	10	-4,-4,-4	43,42,47	-2,-2,-2	32,33,33	-2,-2,-2	18,17,16

Table 3. Sampling: uniform; $N = 100000$; density=800; $E \approx 0.012856$.
Computer: 32 bit i686, 2.8GHz Xeon, 2GB RAM, Red Hat 8.0. g++: no optimization.
Parameters for Morse: $\tau = 0$, $T = \infty$, approach=lazy.

True $\chi = -4$. 5x5x5 cube with 3 tunnels meeting in the middle								
			Incl-excl		Reeb graph		Morse	
δ	#dir 2D	#dir 3D	$\overline{\chi}$	Time(s)	$\overline{\chi}$	Time(s)	$\overline{\chi}$	Time(s)
0.5	5	10	-4,-4	27,26	-4,-4	18,18	-4,-4	10,9

Table 4. Sampling: uniform; $N = 100000$, density=800, $E \approx 0.012856$.
Computer: 32 bit i686, 2.8 GHz Xeon, 2GB RAM, Red Hat 8.0. g++ -O3.
Parameters for Morse: $\tau = 2\delta$, $T = 4\delta$, approach=strict.

5.2.3 Lattice Points in the Cube with 125 Holes

Our next trial was a cube $[0, 104]^3$ with 125 non-overlapping spherical voids of radius 6 in it. The point distribution is $\mathbb{Z}^3$. The minimal distance between balls is 9.

True $\chi = 126$. Cube $[0, 104]^3$ with 125 disjoint bubbles of radius 6.						
			Incl/excl		Reeb graph	
δ	#dir 2D	#dir 3D	$\overline{\chi}$	Time(s)	$\overline{\chi}$	Time(s)
2.4	5	5	126,132	893	125,126	506
2.6	5	5	126,123	876	126,125	485
2.7	5	5	127,125	809	126,127	454

Table 5. Sampling: $\mathbb{Z}^3$; $N = 1045750$; density=1; $\mathfrak{f}(K) = 4.5$.
Computer: 32 bit PC, 2.26 GHz Pentium 4, 1GB RAM, Windows XP Pro.
All times are averaged over two trials.
Morse 2D subroutine was not used, as it was being rewritten at the time of these trials.

True $\chi = 126$. Cube $[0, 104]^3$ with 125 disjoint bubbles of radius 6.								
			Incl-excl		Reeb graph		Morse	
δ	#dir 2D	#dir 3D	$\overline{\chi}$	Time(s)	$\overline{\chi}$	Time(s)	$\overline{\chi}$	Time(s)
2.4	5	5	126	380	126	250	126	135
2.6	1	1	128,126	20,21	125,126	18,17	120,130	12,13
2.6	10	1	125,124	131,134	124,124	88,88	126,123	46,42
2.6	1	10	131,135	211,210	127,125	165,170	125,125	127,121
2.6	5	10	121,127	713,745	125,125	497,496	127,126	260,250
2.7	5	5	126,128	350,355	126,126	240,239	126	129

Table 6. Sampling: $\mathbb{Z}^3$; $N = 1045750$; density=1; $\mathfrak{f}(K) = 4.5$.
Computer: 32 bit i686, 4GB, RAID 1, 2.8GHz Xeon, Red Hat 8.0. g++ -O3.
Parameters for Morse: $\tau = 0$, $T = \infty$, approach=strict.

True $\chi = 126$. Cube $[0,104]^3$ with 125 disjoint bubbles of radius 6.								
			Incl-excl		Reeb graph		Morse	
δ	dir 2D	dir 3D	$\overline{\chi}$	Time(s)	$\overline{\chi}$	Time(s)	$\overline{\chi}$	Time(s)
1.9	$\mathbf{e}_j$ $(j \neq i)$	$\mathbf{e}_i$	57	26	57	19	126	13
2.0	(1,0,1)	(1,1,1)			126	19	126	13
2.0	$\mathbf{e}_j$ $(j \neq i)$	$\mathbf{e}_i$	126	25	126	18	126	12
2.1	$\mathbf{e}_j$ $(j \neq i)$	$\mathbf{e}_i$	126	23	126	18	126	11
2.1	(1,0,1)	(1,1,1)	19984*	25	126	19	120	13
2.4	$\mathbf{e}_j$ $(j \neq i)$	$\mathbf{e}_i$	126	21	126	15	126	10
2.6	$\mathbf{e}_j$ $(j \neq i)$	$\mathbf{e}_i$	126	18	126	12	126	8
3.2	$\mathbf{e}_j$ $(j \neq i)$	$\mathbf{e}_i$	126	13	126	10	126	7
3.2	$\mathbf{e}_j + \mathbf{e}_k$ $(j,k \neq i)$	$\mathbf{e}_i$	124	17	137	13	121	9
6.0	$\mathbf{e}_j + \mathbf{e}_k$ $(j,k \neq i)$	$\mathbf{e}_i$	126	10	126	7	126	5
6.0	$\mathbf{e}_j$ $(j \neq i)$	$\mathbf{e}_i$	76	7	126	6	76	4
6.05	$\mathbf{e}_j + \mathbf{e}_k$ $(j,k \neq i)$	$\mathbf{e}_i$	126	10	106	7	106	5
6.05	$\mathbf{e}_j$ $(j \neq i)$	$\mathbf{e}_i$	76	8	76	5	46	4
6.1	$\mathbf{e}_j + \mathbf{e}_k$ $(j,k \neq i)$	$\mathbf{e}_i$	126	9	106	7	106	5
7.0	$\mathbf{e}_j + \mathbf{e}_k$ $(j,k \neq i)$	$\mathbf{e}_i$	126	8	66	6	66	4
8.11	(1,0,1)	(1,1,1)	126	7	17	5	21	3
8.11175	(1,0,1)	(1,1,1)	125	7	17	5	17	3

Table 7. Sampling: $\mathbb{Z}^3$; $N = 1045750$; density=1; $\mathfrak{f}(K) = 4.5$.
Computer: 32 bit i686, 4GB, RAID 1, 2.8GHz Xeon, Red Hat 8.0. g++ -O3.
Parameters for Morse: $\tau = 0$, $T = \infty$, approach=strict.
*:Value 19984 in row 5 is not an error – recall that Incl-Excl uses slices of width $\delta/2$.

The following table gives some statistical information on errors for the inclusion-exclusion method applied to lattice point data for the cube with 125 holes. Each line corresponds to the average of five estimates of χ, each obtained by using one direction in space and five random directions in 2D (for each direction in 3D the five random directions in 2D have been generated anew). The time column has the total time used for computing all five estimates and averaging them.

True $\chi = 126$. Cube $[0,104]^3$ with 125 disjoint bubbles of radius 6. $\delta = 2.6$ was used everywhere. 2D algorithm: Morse.							
$\overline{\chi}$	mean	max	min	StDev	time	τ	T
126	126.2	129	124	2.17	512	0.65	10.4
126	125.8	131	119	4.44	630	1.3	10.4
126	125.6	130	122	2.97	538	0	∞
126	126	133	119	5.70	421	1.6	12.8
126	126.2	131	121	4.21	424	0	∞

Table 8. Sampling: $\mathbb{Z}^3$; $N = 1045750$; density=1; $\mathfrak{f}(K) = 4.5$.
Computer: 32 bit i686, 4GB, RAID 1, 2.8GHz Xeon, Red Hat 8.0. g++: no optimization.
Morse approach=strict.

5.2.4 Cube with 18 Spherical Holes

Our next trial was a cube $[0, 10]^3$ with 18 non-overlapping spherical voids of radius 1. The point distribution was uniform with density of about 106.

True $\chi = 19$. Cube $[0, 10]^3$ with 18 disjoint bubbles of radius 1. $\delta = 0.8$ was used in all trials							
		Incl-excl		Reeb graph		Morse	
#dir 2D	#dir 3D	$\overline{\chi}$	Time(s)	$\overline{\chi}$	Time(s)	$\overline{\chi}$	Time(s)
1	1	20,18	1,1	20,19,14,22	1,1	19,19	< 0.5,1
10	1	19,19	6,6	18,20,17,22	4,4,4,4	19,19	2,2
1	10	19,19	8,8	18,19,17,19	7,7,7,7	19,20	5,5
5	10	19,19	31,31	20,18,17,19	21,21,21,21	19,19	11,11

Table 9. Sampling: uniform; $N = 100000$; density=105.994; E=0.117; $\mathfrak{f}(K) = 0.5$. Computer: 32 bit i686, 4GB, RAID 1, 2.8GHz Xeon, Red Hat 8.0; g++ -O3. Parameters for Morse: $\tau = 0$, $T = \infty$, approach=strict.

5.3 Analysis of Experimental Results

It appears that in choosing optimal sensitivity parameters, such as δ, τ, and T, different approaches should be taken for lattice and uniform distributions. For the lattice distribution δ can be chosen much smaller than for the uniform one. In practice, δ as small as 1.9 can be used for $\mathbb{Z}^3$, if the Morse algorithm is chosen for 2D. For the uniform distributions δ should be at least 6 times larger than the expected distance to the nearest neighbor. Randomization of the direction of integration is very important for uniform data. To the contrary, for the $\mathbb{Z}^3$ point data randomization is not helpful at all. *If δ is close to 2, then the best directions of integration, both in terms of accuracy and speed, are along the coordinate axes, for both 2D and 3D routines.*

$\mathbb{Z}^n$ Samplings

For $\mathbb{Z}^2$ and $\mathbb{Z}^3$ samplings we observed the following. When coordinate axes have been chosen as directions for integration, *using just one direction for each dimension always produced correct results no matter what algorithm was used, for as along as δ satisfied: $2 \leq \delta \leq \mathfrak{f}(K)$. When δ is known to be close to $\mathfrak{f}(K)$, the coordinate axes are no longer the best bet for the directions of integration – in this case the best 3D direction is (1,1,1), and the best choice of the corresponding 2D direction is (1,0,-1).* Optimal choice of directions for integration is also related to the choice of the 2D algorithm. If $\delta \in [2, 2.4]$, the Reeb graph 2D routine is most tolerant to the change of direction of integration; in fact, it produced correct results no matter which directions have been chosen. If there is a chance that $\mathfrak{f}(K)$, the local feature size, is close to the chosen δ and $\delta \geq 4$, then the 2D algorithm of choice should be Inclusion-Exclusion, since it produces correct results even when δ is larger than $\mathfrak{f}(K)$. When $\mathfrak{f}(K) \approx 1$ (but still $\mathfrak{f}(K) > 1$) the only 2D algorithm that has a chance of correct performance is the Morse routine. It should then be used with $\delta \approx 1.9 - 2$ and (1,1,1) and (1,0,-1) as 3D and 2D directions respectively.

Taking 5 random directions in the plane and 5 in the space guarantees, in practice, the correct estimate of Euler characteristic for $\mathbb{Z}^3$ if δ lies between 2.65 and $\mathfrak{f}(K)$, the local feature size of K.

It is meaningless to use the number of strips that exceeds the number of points. Under the assumption that the number of strips does not exceed than the number of points, the theoretical time-performance of all 2D algorithms is $O(N \log N)$. The same holds for the inclusion-exclusion algorithm in 3D. However, although theoretically the performance of the algorithms does not depend on the strip width, practically, for sampling sizes $N \leq 10^7$, the running time scales linearly with $1/\delta$. If the strip width is fixed, then the Morse (sweeping line) algorithm appears to be the fastest for both lattice and uniform distributions; however, its accuracy seems to be the least stable for uniform data. The inclusion-exclusion 2D algorithm seems to be the most precise for the uniform data, but also the slowest for any type of data.

6 Conclusion

The algorithms presented above offer promising approaches to the estimation of the Euler characteristic of a body K based on a finite set S of partial data. Progress continues on the improvement of implementation and estimation of error bounds for the these algorithms. The algorithms are also being combined with the Cauchy-Crofton formula of Theorem 6 to provide estimates for perimeter (in 2D) and surface area (in 3D) for a body K based on partial data. More generally, these numerical algorithms for *Euler integration* will be combined with traditional algorithms for standard numerical (Lebesgue) integration to provide estimators for mean width, projections, quermassintegrals, and related geometric functionals, by averaging the Euler characteristic of slices of a polytope by flats (or other convenient sets) [KR97, San76]. Hence, these algorithms will lead in turn to techniques for estimating a broad family of geometric functionals based on partial data.

References

[BBCK03] D. K. Blandford, G. E. Blelloch, D. E. Cardoze, and C. Kadow. Compact representations of simplicial meshes in two and three dimensions. In *Proceedings, 12th International Meshing Roundtable*, pages 135–146, 2003.

[DGG03] T. K. Dey, J. Giesen, and S. Goswami. Shape segmentation and matching with flow discretization. In J.-R. Sack F. Dehne, editor, *Proc. Workshop Algorithms Data Strucutres (WADS 03)*, volume LNCS 2748, pages 25–36. M. Smid Eds., 2003.

[DGH01] T. K. Dey, J. Giesen, and J. Hudson. Delaunay based shape reconstruction from large data. In *Proc. IEEE Symposium in Parallel and Large Data Visualization and Graphics*, pages 19–27, 2001.

[Ede99] H. Edelsbrunner. Deformable smooth surface design. *Discrete Comput. Geom.*, 21:87–115, 1999.

[EHMP04] H. Edelsbrunner, J. Harer, A. Mascarenhas, and V. Pascucci. Time-varying reeb graphs for continuous space-time data. In *Proc. of the 20th Annual Symposium on Computational Geometry, Brooklyn*, 2004.

[EJS02] D. Evans, A. J. Jones, and W. M. Schmidt. Asymptotic moments of near neighbour distance distributions. *Proc. Roy. Soc. Lond. Series A*, 458(2028):2839–2849, 2002.

[JB98] M.-F. Roy J. Bochnak, M. Coste. *Real Algebraic Geometry*. Springer-Verlag, 1998.

[KR97] D. Klain and G.-C. Rota. *Introduction to Geometric Probability*. Cambridge University Press, New York, 1997.

[Loj93] S. Lojasiewicz. Sur la géométrie semi- et sous-analytique. *Ann. Inst. Fourier*, 43(4), 1993.

[Mun84] J. Munkres. *Elements of Algebraic Topology*. Benjamin/Cummings, Menlo Park, CA, 1984.

[PM98] A. G. Percus and O.C. Martin. Scaling universalities of kth nearest neighbor distances on closed manifolds. *Adv. Appl. Math.*, 21:424436, 1998.

[Ree46] G. Reeb. Sur les points singuliers d'une forme de pfaff completement integrable ou d'un fonction numerique. *Comptes Rendus Acad. Sciences Paris*, 222:847–849, 1946.

[San76] L. A. Santaló. *Integral Geometry and Geometric Probability*. Addison-Wesley, Reading, MA, 1976.

[Vir88] O. Viro. Some integral calculus based on Euler characteristic. *Lecture notes in Math.*, 1346:127–138, 1988.

[WHDS04] Z. Wood, H. Hoppe, M. Desbrun, and P. Schröder. Removing excess topology from isosurfaces. *ACM Transactions on Graphics (TOG)*, 23(2):190–208, 2004.