A Parameter-Free Normal Estimator on Digital Surfaces

Aude Marêché, Isabelle Debled-Rennesson, Fabien Feschet, Phuc Ngo

A Parameter-Free Normal Estimator on Digital Surfaces

Aude Marêché[1]*, Isabelle Debled-Rennesson[1], Fabien Feschet[2], and Phuc Ngo[1]

[1] Université de Lorraine, CNRS, LORIA, F-54000 Nancy, France
{aude.mareche, isabelle.debled-rennesson, hoai-diem-phuc.ngo}@loria.fr
[2] Université Clermont Auvergne, CNRS, ENSMSE, LIMOS, F-63000 Clermont-Ferrand, France
fabien.feschet@u-auvergne.fr

Abstract. The processing of 3D digital objects often requires the computation and analysis of their geometrical features. The normal vectors of the object's surface in particular provide important information used in image processing applications. We present in this paper a new method for the estimation of normal vectors on the surface of a 3D digital object. It is both local and parameter-free. The proposed method involves the study of neighborhoods around points using planar sectors. Experimental evaluations using multigrid approaches show that it is both faster and more robust than state-of-the-art methods in the field, while being of comparable accuracy.

Keywords: Digital Geometry · Digital plane · Local surface analysis · Normal estimation

1 Introduction

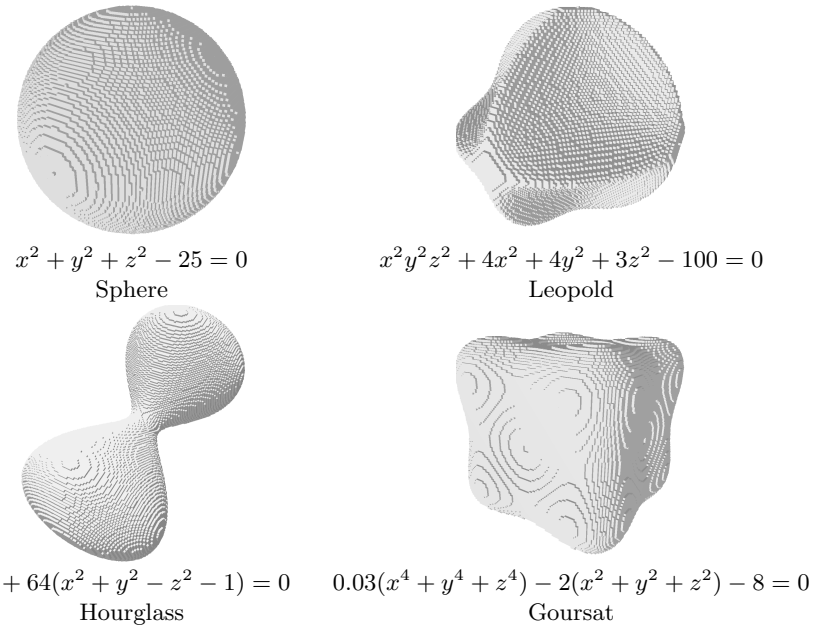
Digital objects are involved in many applications of 3D image processing and medical imaging (e.g. shape recognition, feature extraction, shape matching and visualization). In such contexts, geometric characteristics of these objects are often necessary and have been the subject of many studies. More particularly, estimating differential quantities, such as normal vectors, on the shape boundary typically proves to be a very useful tool for analyzing the local geometry of digital objects.

Several methods have been proposed for normal estimation in the field of Digital Geometry. In [3], Charrier and Lachaud proposed a notion of maximal planes at a given scale to cover a digital surface. Such planes serve as local tangent planes and are used to estimate the normal vector at each point on the surface. The scale value allows to handle the presence of noise on the surface. Then, they used those planes to estimate the normal vector at each point on the surface. Lachaud et al. presented in [9] another approach, called *plane probing* algorithm, for estimating normals. More precisely, the method extends a tetrahedron's neighborhood around a point on a digital surface. Such neighborhood

describes the characteristics of a digital plane among those containing the point under consideration.

In this paper, we propose a new approach to the estimation of the normal vector at each point of a digital surface by structuring the local neighborhood into planar sectors. This enables a more refined analysis of the neighborhood of a point. More particularly, such neighborhoods can be considered as locally representative of the shape, allowing for the estimation of the normal vector at each point.

The proposed method is evaluated and compared with state-of-the-art approaches. The evaluation of differential characteristic estimators is done by showing their *multi-grid convergence* [4]. It consists in testing the estimators on 3D digital objects which are the results of the *digitization* of continuous Euclidean shapes (such as the ones shown Fig. 1). These and other digital objects mentioned in this paper were obtained using Object Boundary Quantization [8], i.e. by computing the intersection of an Euclidean shape with the cubical grid at different resolution. The size of a grid, called the *digitization step* of the digitized shape and denoted by h, is the distance between two consecutive cells of the grid. For example, in Figure 1, the four Euclidean shapes were digitized with $h = 0.25$. To demonstrate *multi-grid convergence* of a normal estimator, its estimations must be closer to the real value on the Euclidean shape the smaller h is.



$x^2 + y^2 + z^2 - 25 = 0$
Sphere

$x^2y^2z^2 + 4x^2 + 4y^2 + 3z^2 - 100 = 0$
Leopold

$z^4 + 64(x^2 + y^2 - z^2 - 1) = 0$
Hourglass

$0.03(x^4 + y^4 + z^4) - 2(x^2 + y^2 + z^2) - 8 = 0$
Goursat

Fig. 1: The digital surfaces considered in this paper and the polynomial equations defining the underlying Euclidean shapes.

This article is organized as follows: Section 2 provides some definitions used in this paper as well as a state of the art about normal estimators on digital objects. The proposed approach to obtain a new parameter-free normal estimator, based on the neighborhood structuring, is described in Section 3. In Section 4, we propose an experimental evaluation of this estimator.

2 Preliminaries

This section first recalls some definitions that are used in this paper, followed by the state of the art presentation.

2.1 General Definitions

A 2D digital image can be represented as a grid of square cells whose center are points of integer coordinates (i.e. in $\mathbb{Z}^2$), usually called *pixels*. This paper deals with 3D objects. Similarly, they can be defined in the digital space of $\mathbb{Z}^3$ and with a grid of cubic cells. These cubes are *digital points* of $\mathbb{Z}^3$, referred to as *voxels*, and their faces can be referred to as *surfels* (shown Figure 2); for ease of notation, we denote the set of surfels of a given voxel v by $\text{Surfs}(v)$.

Two voxels are α-*connected*, or α-*adjacent*, if they share a surfel, an edge, or a vertex, with $\alpha = 6$, 18, or 26 respectively to each of these three cases (as illustrated in Fig. 2). For a given set of voxels $X \subset \mathbb{Z}^3$, an α-*path* between two voxels $a, b \in X$ is a sequence of α-connected voxels of X joining a to b. If an α-path exists for every $(a, b) \in X$, then X is said to be α-connected.

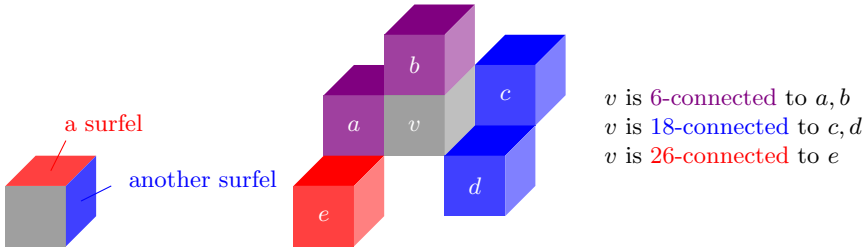


Fig. 2: Types of adjacencies on voxels.

We study digital objects O which are 6-connected finite subsets of $\mathbb{Z}^3$. Let such an object's *surface* be $S_O = \{s \in \text{Surfs}(p_1) \cap \text{Surfs}(p_2) \mid p_1 \in O, p_2 \in \mathbb{Z}^3 \setminus O\}$ be the set of surfels forming the interface between O and its complement in $\mathbb{Z}^3$. Let $V_O = \{p \in O \mid \exists s \in Surfs(p) \cap S_O\}$ be the subset of voxels of O for which at least one surfel belongs to S_O, or in other words the voxels that are visible from the outside of the object O (see Fig. 1).

Let us define a *digital plane* as the set of voxels $x \in \mathbb{Z}^3$ such that

$$\mu \leq \langle x, n \rangle < \mu + \omega \tag{1}$$

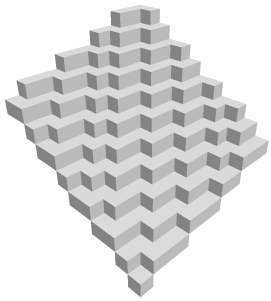


Fig. 3: A piece of the naive digital plane with characteristics $n = (4, 7, 11)$ and $\mu = 2$.

where $\mu \in \mathbb{Z}$ is the digital plane's offset from the origin of the space, $\omega \in \mathbb{Z}$ is its thickness, and $n \in \mathbb{Z}^3$ is the normal vector to the digital plane. In other words, a digital plane is the set of digital points located between two parallel real planes, with n being both the normal vector to the digital plane and to the two *supporting* real planes. A digital plane for which $\omega = \max(|a|, |b|, |c|)$ is said to be *naive*; these naive digital planes are the thinnest digital planes retaining 18-connectedness without having 6-connected holes [1]. While an infinite digital plane has unique characteristics, an infinite set of possible characteristics can be associated to a finite piece of a digital plane. Digital plane recognition algorithms from the literature thus return one tuple of characteristics n, μ and ω among the possible ones.

2.2 State of the Art

Digital Plane Recognition We recall in this paragraph two methods to recognize pieces of naive digital planes.

The first one was proposed by Gérard et al. [7], called *Chord algorithm*. It uses the chord set of the considered voxels, and more precisely its thickness. If it is thin enough, this guarantees the existence of at least one digital plane containing the considered voxels. This method then estimates the characteristics of this digital plane, with an empirical behavior for the whole process that is quasi-linear.

The other method is COBA, short for *Convex Optimization-Based Algorithm*, from Charrier and Buzer [2]. It consists in considering digital plane recognition as a feasibility problem on a convex function, corresponding to the distance between the two supporting hyperplanes of the possible digital plane containing the considered set of voxels. The use of cuts on the solution space then allows the authors to obtain a linear worst-case complexity, and an empirical computation time that is comparable to that of the Chord algorithm.[3]

[3] See `https://www.dgtal.org/doc/stable/modulePlaneRecognition.html#modulePlaneRecognition_sec5` for more details.

Digital Normal Estimation By using one of the previous digital plane recognition methods on a planar neighborhood of a point on a digital surface, a normal estimation can be deduced from the characteristics of the recognized plane. Charrier and Lachaud [3] proposed a notion of maximal planes at a given scale covering a digital surface. It can also be used to estimate the normal vector at each point on the surface.

Another approach to the problem of digital normal estimation is that of *plane probing* algorithms (PP), introduced by Lachaud et al. in [9]. It gives a normal vector estimation by extending around a point on a digital surface a tetrahedron that describes the characteristics of a digital plane. The choices made when extending this tetrahedron's neighborhood allow this approach to select the correct digital plane among the ones containing the considered voxels and to compute, when used on a surface contained by a digital plane, its exact normal.

Cœurjolly et al. presented in [5] a normal estimation method using the notion of *integral invariants* (II) to produce curvature and normal estimators with the property of multi-grid convergence on sufficiently smooth (C^3) surfaces, with a theoretical convergence speed of $O(h^{\frac{2}{3}})$. This property guarantees that the estimation computed at a given point on a digital surface gets closer to the expected real value on the corresponding Euclidean surface when the digitization step h gets smaller and the digitization itself gets finer. This method consists in applying a kernel of a given size on the digital surface and then computing integral quantities on the intersection between this kernel and the digital surface, from which the desired information can then be deduced. It is considered the state of the art in Digital Geometry.

Since a meaningful comparison with the normal vector estimation methods used for meshes would require a proper mapping between meshes digital surfaces, it will be left for an extended version of this paper.

3 Proposed Approach

In this section, we describe our local approach for the estimation of normal vectors on a digital surface following three steps:

1. first, we build a maximal planar circular neighborhood around a voxel on the surface (Figure 4);
2. then we extend this neighborhood in order to recover local planarity information by constructing planar angular sectors (Figure 5);
3. finally, we compute the normal estimation by averaging the normal vectors to the planar sectors (Section 3.2).

3.1 Neighborhood Around a Point

Let $p \in V_O$ be the point on the surface of the object at which we want to compute a normal estimation. First, we build a *neighborhood* of radius $r \in \mathbb{N}^*$ around p

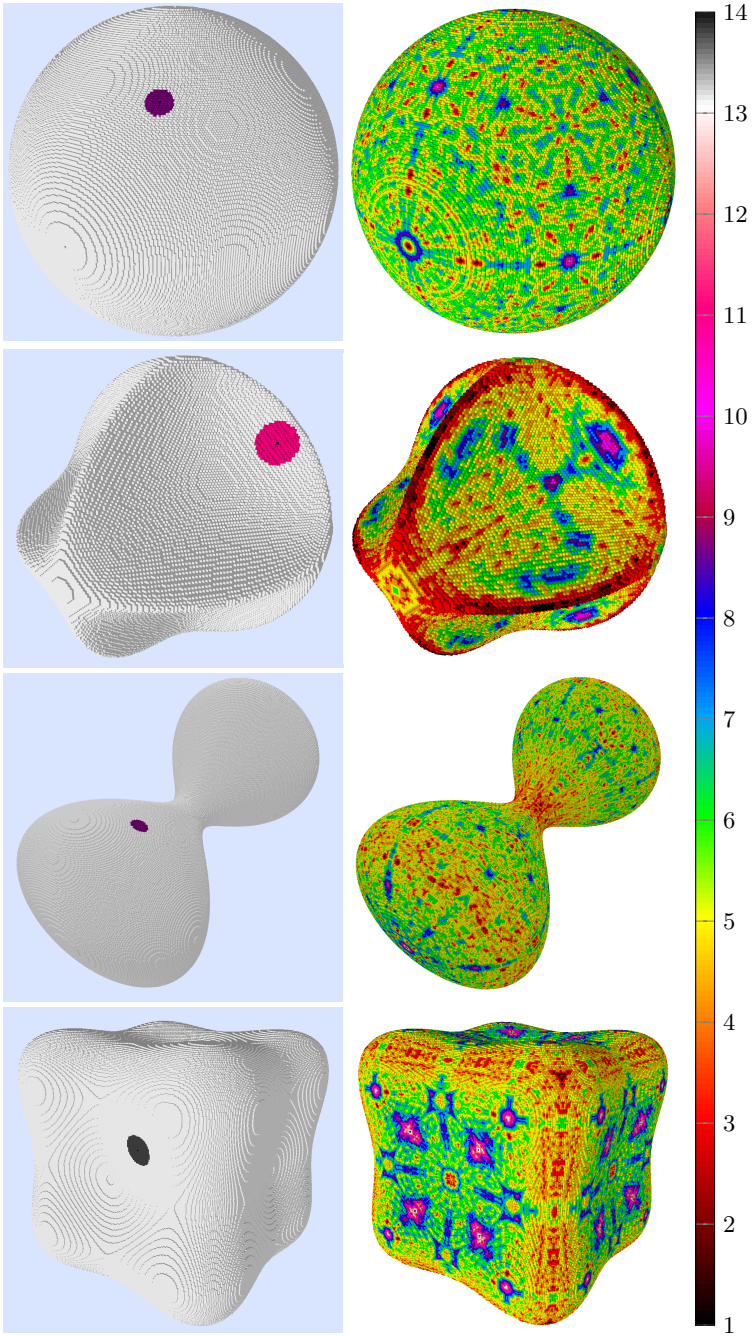
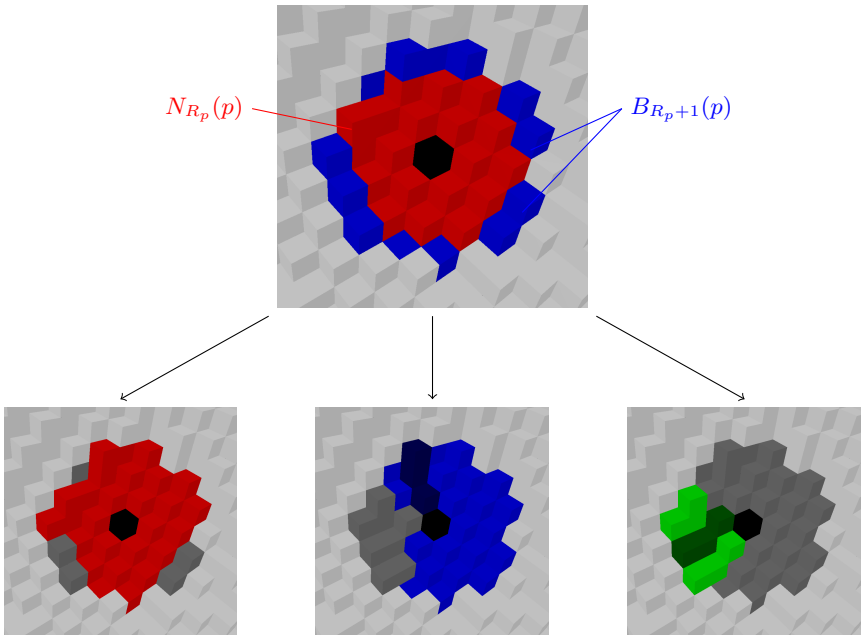


Fig. 4: Examples of some of the largest neighborhoods computed on the objects. The colormap corresponds to the value of R_p for each voxel of the surface.



(a) plane $\mathbf{P}_{R_p}$ (b) first sector (c) second sector

Fig. 5: A voxel p (black) and its neighborhood of size 4 ($R_p = 3$). The bottom pictures show (a) the digital plane $\mathbf{P}_{R_p}$ in red, and in (b) and (c) the two planar sectors constructed from the voxels not belonging to $\mathbf{P}_{R_p}$. The darker-colored voxels in (b, c) show the starting path for the sector.

defined as $N_r(p) = \{q \in V_O \mid d(p,q) \leq r$ and $p,\ q$ are 18-connected in $V_O\}$. 18-connectedness ensures that the neighborhood retains simple connectedness on the surface of the object.

We denote by R_p the biggest radius for which N_{R_p} belongs to a naive digital plane, named $\mathbf{P}_{R_p}$. We compute R_p by incrementing r one by one until either $N_r(p)$ is no longer planar or $N_{r+1}(p) = N_r(p)$, i.e. there are no new voxel to add at this distance. Figure 4 shows examples of planar neighborhoods found on the digital objects studied on this paper, and a distribution map of the different values of R_p on these objects.

3.2 Construction of Planar Angular Sectors

For our normal estimation, we consider the non-planar neighborhood $N_{R_p+1}(p)$; we define its *border* as $B_{R_p+1}(p) = N_{R_p+1}(p) \setminus N_{R_p}(p)$. If $B_{R_p+1}(p)$ is not empty, some of its voxels may belong to $\mathbf{P}_{R_p}$, but not all, as shown Figure 5(a). Those voxels in $B_{R_p+1}(p)$ that do not belong to $\mathbf{P}_{R_p}$ are significant for characterizing the local structure of the digital surface: we therefore use them as the starting points for building planar sectors.

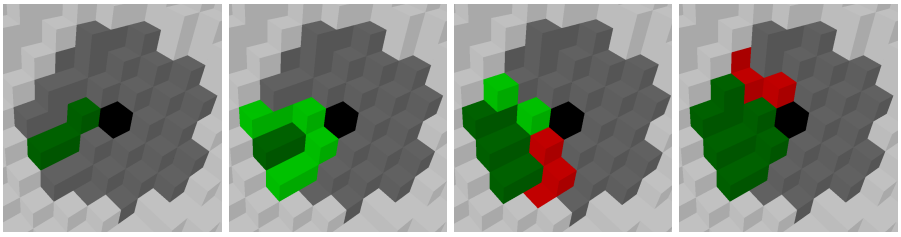


Fig. 6: Step-by-step construction of the second sector from Figure 5. At each step, accepted paths are shown in green, while red paths do not belong to the same digital plane as the rest of the sector and are thus rejected.

Union of Voxel Paths To build a planar sector, we start with a voxel e in $B_{R_p+1}(p)$ that does not belong to $\mathbf{P}_{R_p}$. We compute a 26-path in $N_{R_p+1}(p)$ from e to p, staying close to the line segment $[ep]$ by minimizing the angle between this segment and the one between e and the next voxel of the path at each step (see Fig. 6). 26-paths are preferred to 18-paths for this as they are more likely to stay close to $[ep]$ and to belong to a naive digital plane. The 18-paths are included in the 26-paths, there are thus more possibilities with 26-paths. Such paths are also computed for e's neighbors in $B_{R_p+1}(p)$; if they all belong to the same naive digital plane, we consider their union to be a sector. We then continue to add neighboring paths, alternately on each side of the current sector, until adding the next one would make the sector non-planar. Figure 6 (right) shows an example of result from this process.

We repeat this process until every voxel of $B_{R_p+1}(p)$ not in $\mathbf{P}_{R_p}$ has been included in at least one sector. We then consider the set of sectors built in this manner, to which we add the set of voxels belonging to the plane $\mathbf{P}_{R_p}$, as an additional sector.

Normal Estimation The sectors obtained through the previous step are all planar. We can therefore associate to each of them the corresponding normal vector to their digital plane. By computing an average over these vectors, we obtain an estimation of the normal vector at p. This average is computed by normalizing the normal vectors and projecting them on the surface of the unit sphere. We then use the method from [10] to build a regular grid on the unit sphere, with an empirical precision of approximately 3.10^{-5}, in order to approximate the geodesic barycenter of the normal vectors to the sectors. Using this method, we can thus compute a normal estimation for every voxel on the surface of a 3D digital object.

4 Experimental Results

3D Digital Objects Studied Figure 1 shows the four digital surfaces that were considered in the experiments, constructed from four continuous Euclidean

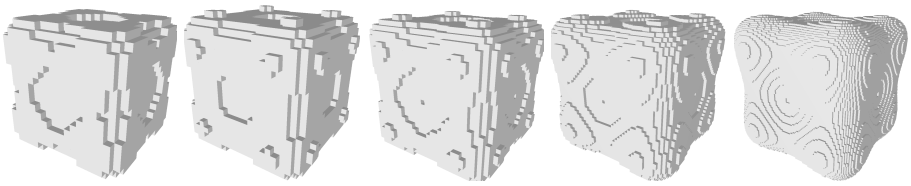


Fig. 7: The digital surfaces resulting from the digitization of Goursat using decreasing digitization steps h, from $h = 1$ to $h = 0.2$.

Table 1: Number of voxels in the digitizations of the four shapes using decreasing digitization steps.

h	Sphere	Leopold	Hourglass	Goursat
1	222	194	254	1 634
0.8	354	298	414	2 546
0.6	654	570	778	4 614
0.4	1 550	1 406	1 830	10 590
0.2	6 342	6 062	7 546	42 842
0.1	25 746	24 874	30 754	172 686
0.08	40 442	38 914	48 190	270 206
0.06	71 894	69 374	86 038	480 906
0.04	162 290	156 586	194 166	1 083 242

shapes. The Sphere and the Hourglass are convex; Goursat is convex with slightly concave areas, while Leopold is mostly concave with thin edges. As mentioned in the introduction of this paper, these digital surfaces were obtained using Object Boundary Quantization [8]. This allows to generate the digital object corresponding to a real shape defined by a polynomial equation at a chosen resolution. This is done by using grids of different sizes (h) which intersect the real shape. Consequently, we generate digital surfaces of increasing precision, and use the normal vectors to the underlying real shape as ground truth. As a result, the smaller h is, the bigger the resulting digitized object will be, or in other words the better its resolution will be. Smaller h will thus result in digital objects that are closer to the underlying Euclidean shape (see Figure 7). We will show in the experiments the multi-grid convergence behavior of our estimator: the smaller h is, the more correct the estimate of the normals is and therefore the less error there is.

Computation Time Analysis As Table 1 shows, the number of voxels of a digitized shape increases rapidly as h decreases. The normal estimation methods we considered in this paper use a local approach; thanks to this, the time they need to process an entire digital surface stays relatively low. However, as h decreases and the digital objects get closer to the real shapes, these methods take into consideration neighborhoods of increasing size, and each voxel takes more time to process. Table 2 contains, for each method, the computational

Table 2: Number of voxels processed per minute for each object and digitization step by each method.

Sphere

h	ours	r	II	PP
1	all	all	all	4 440
0.8	17 700	all	all	2 950
0.6	13 080	32 700	32 700	1 557
0.4	9 118	22 142	19 374	775
0.2	6 342	12 196	6 894	181
0.1	4 291	8 582	1 716	47
0.08	3 111	6 740	1 444	–
0.06	2 568	4 793	549	–
0.04	2 497	3 453	–	–

Leopold

h	ours	r	II	PP
1	all	all	all	all
0.8	all	all	all	all
0.6	28 500	28 500	28 500	all
0.4	17 574	28 119	20 085	703
0.2	8 304	15 155	7 772	164
0.1	4 975	8 291	2 073	45
0.08	3 538	6 486	1 297	–
0.06	2 775	4 955	559	–
0.04	2 303	4 121	–	–

Hourglass

h	ours	r	II	PP
1	all	all	all	5 080
0.8	20 700	all	all	2 435
0.6	11 114	25 933	38 900	1 496
0.4	10 765	26 141	18 300	610
0.2	7 546	14 511	7 546	145
0.1	4 393	10 251	1 464	42
0.08	3 707	9 638	927	–
0.06	3 073	7 170	574	–
0.04	1 981	5 110	–	–

Goursat

h	ours	r	II	PP
1	8 170	23 341	81 699	1 634
0.8	7 274	19 585	50 919	509
0.6	7 098	17 089	46 140	256
0.4	5 295	13 237	22 531	163
0.2	3 570	7 140	8 568	40
0.1	2 186	4 667	1 919	–
0.08	1 851	4 503	1 518	–

speed we observed in our experiments, in terms of voxels processed per minute, using the reference implementations provided by the authors (see Section 4). It shows that our method scaled better than II and PP on the four digital surfaces; as a result, and in order to keep computation times down, we did not pursue the experiments when these two methods became notably slow compared to ours.

Implementation Details The experiments presented here were carried out using DGtal [6], an open-source C++ library providing data structures and algorithms for Digital Geometry applications. It provides reference implementations by their authors of the normal estimators we compare our method to. The algorithm we used for the recognition of naive digital plane pieces, the Chord algorithm (see Sect. 2.2), is also implemented in DGtal, as well as many others.

Results Figure 8 shows the results obtained on each digital surface. On the left-hand side, the graph represents the average computation time in minutes of each normal estimator, with the number of voxels of the digital surface by way of comparison. The right-hand side graphs show, for each method, the mean error between its normal estimations and the ground truth. This error is computed as the geodesic distance between the normalized vectors projected on the unit

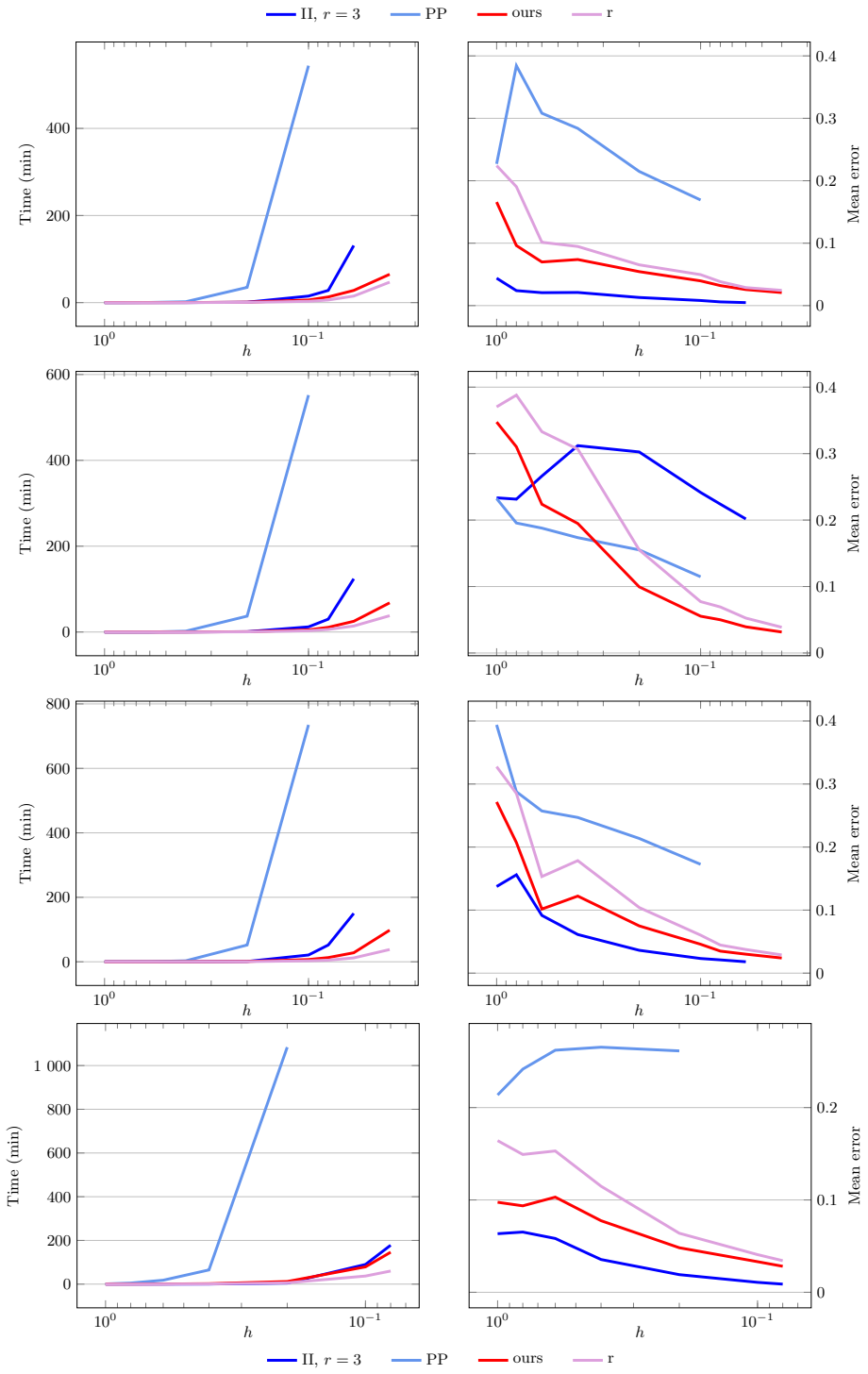


Fig. 8: Computation times and mean errors for the four objects in relation to h (from top to bottom the Sphere, Leopold, the Hourglass, and Goursat).

sphere and averaged over the whole digital surface. In the legend, the acronyms respectively refer to the results obtained using the Integral Invariants method (II), the plane probing method (PP), the method presented here (ours), and the normal vectors to the digital planes $\mathbf{P}_{R_p}$ at each point p of the surface (r).

The digitization process sometimes results in voxel artifacts, for which there is no naive digital plane containing the neighborhood (i.e. $R_p = 0$). In these cases, the average of the normal vectors to the visible surfels is used for our method and (r).

In the following, we will describe the results obtained by our normal estimator compared with the normal vectors to the digital planes $\mathbf{P}_{\mathbf{R_p}}$, the plane probing method, and the Integral Invariants method.

- The figures show that our estimations have a similar behavior to the normal vectors given by the digital planes (r). Our method however has a consistently smaller mean error, for an execution time that is quite similar. Moreover, the estimations are improved for high digitization steps in particular, meaning when the digital surface is less precisely fitted to the real shape.
- The plane probing estimator shows an overall worse performance than the other estimators (which was predictable, see 2.2). It is also much slower than both our method and Integral Invariants. Nonetheless, it shows interesting behavior on Leopold, where it outperforms other estimators, including ours, when h is big. This is mostly due to the fact that Leopold is both the smallest shape out of the ones considered here, with a digital surface of only 194 voxels when $h = 1$, and is the most concave shape. This is however only true for the biggest h, as our method quickly "catches up" to the plane probing estimator, and gets significantly better as h decreases. Moreover, and as the graphs for other shapes show, the behavior of the plane probing method is quite erratic on convex shapes, and we outperform it in all other cases.
- On the three convex shapes (the sphere, the hourglass and Goursat), the Integral Invariants estimations have the lowest error of the methods we compared, as was expected. However, it can be noted that our estimator behaves in a very similar way on these shapes and our mean error stays close to that of II, especially on the hourglass shape on one hand, and in comparison to the plane probing estimator on the other hand. Integral Invariants also shows signs of erratic behavior on Leopold when the digitization step is big, before appearing to stabilize, but with a relatively high error compared to the other methods. This shows that while our method performs slightly worse in regard to precision, it has a more consistent behavior over different surfaces. Another important point is that II is significantly slower than our method when h increases.

These experiments show that our new method for normal estimation provides an alternative to state-of-the-art methods, using only the information given by digital plane pieces in a local neighborhood. While not as precise as Integral Invariants on convex shapes, it is both faster and more robust on concave shapes.

Our method also displays improving precision when the shapes' digitizations get finer, which hints towards a possible multi-grid convergence. Moreover, this algorithm is easily parallelized, as each voxel is processed independently.

5 Conclusion

We have presented in this paper a new method for the estimation of normal vectors on digital surfaces. This new method is local and parameter-free, and gives results that are comparable to the state of the art in terms of mean error, while being faster and more stable on different surfaces. While the multi-grid convergence of this estimator is not formally proved, we do observe experimental convergence and a similarity of behavior with a state-of-the-art convergent normal estimator. Possible future directions for this work include using thicker digital planes, and exploring the behavior of the method on noisy surfaces. We believe this work can open research directions on the study of local properties of digital surfaces using planar sectors, and we will continue to explore this topic in the future.

References

1. Andres, E., Acharya, R., Sibata, C.: Discrete analytical hyperplanes. Graphical Models and Image Processing **59**(5), 302–309 (1997)
2. Charrier, E., Buzer, L.: An efficient and quasi linear worst-case time algorithm for digital plane recognition. In: Discrete Geometry for Computer Imagery, 14th IAPR International Conference, DGCI. vol. 4992, pp. 346–357 (2008)
3. Charrier, E., Lachaud, J.O.: Maximal planes and multiscale tangential cover of 3d digital objects. In: Combinatorial Image Analysis - 14th International Workshop (IWCIA). vol. 6636, pp. 132–143 (2011)
4. Coeurjolly, D., Flin, F., Teytaud, O., Tougne, L.: Multigrid convergence and surface area estimation. In: Geometry, Morphology, and Computational Imaging, 11th International Workshop on Theoretical Foundations of Computer Vision. Lecture Notes in Computer Science, vol. 2616 (2002)
5. Coeurjolly, D., Lachaud, J.O., Levallois, J.: Integral based curvature estimators in digital geometry. In: Discrete Geometry for Computer Imagery - 17th IAPR International Conference, DGCI. vol. 7749, pp. 215–227 (2013)
6. Dgtal: Digital geometry tools and algorithms library, `http://dgtal.org`
7. Gérard, Y., Debled-Rennesson, I., Zimmermann, P.: An elementary digital plane recognition algorithm. Discrete Applied Mathematics **151**(1-3), 169–183 (2005)
8. Klette, R., Rosenfeld, A.: Digital geometry - geometric methods for digital picture analysis. Morgan Kaufmann (2004)
9. Lachaud, J.O., Provençal, X., Roussillon, T.: An output-sensitive algorithm to compute the normal vector of a digital plane. Theoretical Computer Science **624**, 73–88 (2016)
10. Roşca, D.: New uniform grids on the sphere. Astronomy & Astrophysics **520**, A63 (2010)