HAL
archives-ouvertes.fr

An adaptative filtering method to evalute normal vectors and surface areas of 3d objects. Application to snow images from X-ray tomography

Frédéric Flin, Jean-Bruno Brzoska, David Coeurjolly, Romeu Pieritz, Bernard Lesaffre, Cécile Coléou, Pascal Lamboley, Olivier Teytaud, Gérard Vignoles, Jean-François Delesse

To cite this version:

Frédéric Flin, Jean-Bruno Brzoska, David Coeurjolly, Romeu Pieritz, Bernard Lesaffre, et al.. An adaptative filtering method to evalute normal vectors and surface areas of 3d objects. Application to snow images from X-ray tomography. IEEE Transactions on Image Processing, Institute of Electrical and Electronics Engineers, 2005, 14 (5), pp.585-596. <hal-00185086>

HAL Id: hal-00185086
https://hal.archives-ouvertes.fr/hal-00185086

Submitted on 5 Nov 2007

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Lowres version

Adaptive estimation of normals and surface area for discrete 3D objects: application to snow binary data from X-ray tomography

Frédéric Flin, Jean-Bruno Brzoska*, David Coeurjolly, Romeu André Pieritz, Bernard Lesaffre, Cécile Coléou, Pascal Lamboley, Olivier Teytaud, Gérard Vignoles, Jean-François Delesse

F. Flin, J.B. Brzoska, R.A. Pieritz, B. Lesaffre and C. Coléou are with the Centre d'Etudes de la Neige, Météo-France, 1441 r. de la Piscine, F38406 St. Martin d'Hères cedex (Phone: (33) 4.76.63.79.25, Fax: (33) 4.76.51.53.46, e-mail: first_name.second_name@meteo.fr).

D. Coeurjolly is with the ERIC Laboratory, Université Lumière Lyon 2, 5 av. Pierre-Mendès-France, F69676 Bron cedex (e-mail: dcoeurjo@eric.univ-lyon2.fr).

P. Lamboley is with the DP/Prévi/Compas, Météo-France, 42 av. G. Coriolis, F31057 Toulouse cedex (e-mail: pascal.lamboley@meteo.fr).

O. Teytaud is with the ISC, 67 bd. Pinel, F69675 Bron cedex (e-mail: oteytaud@isc.cnrs.fr).

G. Vignoles is with the LCTS, UMR 5801 CNRS-SNECMA-CEA-Université Bordeaux 1, 351 crs. de la Libération, F33405 Talence cedex (e-mail: vinhola@lcts.u-bordeaux.fr).

J.F. Delesse is with the LaBRI, UMR 5800 CNRS-Université Bordeaux 1, 351 crs. de la Libération, F33405 Talence cedex (e-mail: delesse@labri.u-bordeaux.fr).

Lowres version

EDICS: 2-ANAL, 3-TOMO

Abstract

Estimating the normal vector field on the boundary of discrete 3D objects is essential for rendering and image measurement problems. Most of the existing algorithms do not provide an accurate determination of the normal vector field for shapes that present edges.

We propose here a new and simple computational method in order to obtain accurate results on all types of shapes whatever their local convexity degree is. The presented method is based on the gradient vector field analysis of the object distance map. This vector field is adaptively filtered around each surface voxel using angle and symmetry criteria, so that as many relevant contributions as possible are accounted for. This optimizes the smoothing of digitization effects while preserving relevant details of the processed numerical object.

Thanks to the precise normal field obtained, a projection method can be proposed to derive immediately the surface area from a raw discrete object. An empirical justification of the validity of such an algorithm in the continuous limit is also provided. Some results on simulated data and snow images from X-ray tomography are presented, compared to the Marching Cubes and Convex Hull results and discussed.

Index Terms

adaptive filtering, discrete geometry, distance map, normal vectors, surface area, snow, X-ray tomography.

I. Introduction

Once fallen on the ground, snow undergoes a structure metamorphism governed by local temperature and humidity fields. Local 3D curvature is a governing parameter of metamorphism [1], whereas surface area denotes the ability for a given snow to change its own microstructure. To access these parameters, the normal vector field has to be known all along the digitized surface of the snow sample.

The work presented below was designed for processing grey level images from X-ray absorption microtomography of low-density snow samples that were prepared using a filling medium [2]. Such gray-level data present three different phases (ice, filler and air) along with some noise and phase-contrast effects, and require a specific treatment to be contoured correctly [3]. The starting point of the 3D image analysis is then two-level (binary) volume data.

To obtain the normal vector field, meshing the initial binary discrete image is the most popular approach. Generally, it uses methods related to the Marching Cubes algorithm [4]: the easy derivation of normals from facets is used in many shading algorithms [5]. Although various smoothing and decimation procedures exist [6], the initial accuracy of triangle positioning

Lowres version

is essential. Many imaging scientists have developed normal vector estimation algorithms. A synthesis of the works done until 1992 can be found in [7]. From this date, new approaches considering the 3D neighborhood of the points have appeared. A first approach consists in working on the volume of the image as in [8]. Other algorithms use the surfel (surface element) decomposition of the digitized object [9], then average surfel elementary normals using convenient weightings and neighborhoods [10], [11]. Other interesting approaches consist in considering two-dimensional slices of a 3D image [12], [13].

All these techniques generally provide good and effective results on smooth surfaces such as spheres, tori and infinite plane surfaces. However, they often do not provide an accurate normal vector field for shapes that present edges: if the chosen neighborhood is too small, the corresponding vector would be sensitive to digitization effects. On the other hand, if it is too large, sharp shapes would be smoothed. This problem, discussed in [14], seems to be common to all the existing algorithms.

We propose here a new and simple algorithm to obtain accurate results on all types of shapes whatever their degree of local convexity is. A first approach, presented in [15], was completed to obtain a fully adaptive algorithm: a discrete background distance map of the original 3D image is first constructed. For each voxel of the object, an elementary gradient vector field is then computed. The normal vector of a surface voxel is obtained by summing each gradient vector in a neighborhood which is determined by angular and symmetry criteria. Thanks to these two criteria, the shape and the size of the working neighborhood are automatically adapted to the local geometry of the surface.

As a very precise vector field is computed on each surface voxel of the object, an accurate estimation of the surface area can be obtained directly from this field: when the surfel decomposition of the surface is available, this can be done by summing the projected area of each surfel along the local normal vector [12], [16]. A method applicable to raw voxel data is detailed in the paper.

After a short part where some definitions and basic properties are reminded, the determination of the normals is addressed with the detailed presentation of the adaptive algorithm. The surface area estimation method is then described. For both methods, validations and multigrid convergence tests are provided. An application to snow images from X-ray tomography is finally presented.

Lowres version

II. Definitions and basic properties

A. Basic definitions

In 3D, by analogy to 2D, a point in $\mathbb{Z}^3$ is called a *voxel* and a discrete object is defined as a connected set of voxels using a 6-, 18- or 26-connectivity. Based on this discrete object definition, discrete topological properties can be designed [17]. In the following, we consider a 18-connected discrete object, denoted by $\mathcal{O}$. We also define a discrete surface of such an object as the set of voxels such that one of their 6-neighbors belongs to the complementary of the discrete object, denoted by $\mathcal{S}$. This surface definition proposed by Morgenthaler and Rosenfeld [18] verifies some topological properties as the Jordan-Brouwer theorem that provides for example no holes in the surface. Other discrete surface definitions exist considering for instance a cellular complex decomposition [9] of the object. In this paper, we identify each voxel with its central point.

B. Background distance map

Definition 1: Let us consider two voxels p and q. Let $d(p,q)$ be the distance from p to q. In a numerical object $\mathcal{O}$, the background distance d_b on a point $p \in \mathcal{O}$ is defined as follows:

$$\forall q \notin \mathcal{O}, d_b(p) = \min[d(p,q)] \tag{1}$$

In computer imagery, we only consider integer metrics. Hence, several approaches are possible; some of them consider the distance between two points as the integer displacement (dx, dy) [19], [20]. Another approach consists in using truncated metrics based on a fixed-size mask estimation of the Euclidean metric. These metrics are called chamfer metrics [21], [22] and lead to efficient distance map algorithms.

C. Multigrid convergence

If we formalize the digital surface area estimation problem, we consider the Euclidean 3D-space $\mathcal{E}^3 = (\mathbb{R}^3, d_2)$, d_2 being the euclidean distance. We denote by $\mathcal{S}$ a Jordan surface in $\mathcal{E}^3$ and a digitization process denoted by dig_r, where r denotes the resolution of the grid (the distance between two grid points). Such a digitization process can be the Grid Intersect Quantization scheme (GIQ for short), the Object Boundary Quantization scheme (OBQ for short) or the Background Boundary Quantization scheme (BBQ for short) (see [23] for example of a digitization scheme survey).

Let us consider a digitization-based surface area estimator $E_{\mathcal{A}}$. An important property of this estimator is its multigrid convergence [24]:

Definition 2: Let $\mathcal{S}$ be a Jordan surface in $\mathcal{E}^3$, dig_r be a digitization process and $E_{\mathcal{A}}$ be a

Lowres version

surface area estimator based on $dig_r(\mathcal{S})$. $E_{\mathcal{A}}$ is multigrid convergent with a convergence speed $v(r)$ if and only if there exists r_0 and $r \leq r_0$ such that:

$$|E_{\mathcal{A}}(dig_r(\mathcal{S})) - \mathcal{A}(\mathcal{S})| \leq v(r) \quad \text{and} \quad \lim_{r \to 0} v(r) = 0 \tag{2}$$

In the following, experimental evaluations of both normal vector and surface area estimation are given using this framework. More precisely, digitizations of continuous objects on different grid size are computed and the estimator values are compared to the known Euclidean values on the underlying continuous objects.

III. Normal vector estimation

A. Algorithm: Adaptive Distance Gradient Filtering

To compute normal vectors, we chose a volumic approach. Such an approach is time-consuming, but allows to use all the information contained in the object and gives more accurate results. Moreover, the presented method is simple and can be easily implemented by scientists who are non-familiar with complex data structures. The key idea of the algorithm is to compute the normal vector $\vec{n}(p)$ by averaging the gradient vector orientations of the background distance map on an appropriate neighborhood of a voxel p. In the following, this approach will be named Adaptive Distance Gradient Filtering (ADGF for short).

A.1 Distance map

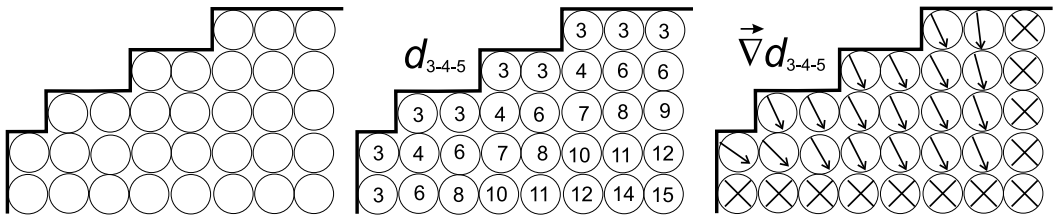
In order to build the background distance map, we chose a two-run iterative chamfer algorithm. In summary, this algorithm increments at each run the discrete distance to the closest already processed voxel of the object. Depending on the size of the used chamfer kernel, some distortion with respect to the Euclidean distance map may occur in deeper parts of the map ; however, this effect is negligible near the surface, i.e. on the first layers of voxels.

For the presented method, we used a second-neighbor chamfer distance $d_{5-7-9-11-12-15}$ (d_5 for short), which seems to be a good compromise between the fastness of the first-neighbor chamfer distance d_{3-4-5} and the accuracy of the Euclidean distance [25], [26].

A.2 Gradient map

For each voxel $p \in \mathcal{O}$, $\vec{\nabla}_{d_b}(p)$ is obtained by using a classical Prewitt first-neighbor mask (Fig. 1). The gradient map obtained is very sensitive to each detail of the distance map. To compute an accurate normal vector field on the surface $\mathcal{S}$ of $\mathcal{O}$, the elementary gradient vectors have to be summed on an appropriate neighborhood.

Lowres version



(a) (b) (c)

Fig. 1. Obtaining the gradient map in 2D. Binary (a), distance map (b) and gradient map (c) images.

A.3 Summation on an adaptive shape neighborhood

A simple summation of the elementary gradient vectors in a fixed spherical neighborhood leads to a bad description of the surface: if the radius of the neighborhood is too small, the corresponding vector would be sensitive to digitization effects. On the other hand, if it is too large, shapes would be smoothed. In this case, acute shapes like the vertices of a cube would become rounded. In other words, finding out the optimal radius for a whole object is a difficult compromise between smoothing enough the digitization steps and preserving its fine details. This approach inevitably leads to a loss of information. In fact, an adaptive search neighborhood is required to take into account the whole geometry of complex shapes.

The main idea of the presented method is to find, for each point p of the surface $\mathcal{S}$, the most convenient 3D neighborhood so that as many relevant contributions as possible would be accounted for. It can be built starting from a spherical neighborhood whose radius is large enough to obtain a normal vector field without digitization effects. To ensure that significant details would not be smoothed, just a part of the gradient vectors belonging to the sphere has to be taken into account.

To decide whether a gradient vector has to be counted or not, it has to be tested in terms of angular and symmetry criteria.

Angular criterion:

Let p be a surface voxel of an object $\mathcal{O}$. If p belongs to a smooth surface, the normal vector orientation does not vary significantly from a voxel to its closest neighboring voxel. In this case, all gradient vectors in a small spherical neighborhood around p are relevant for the normal vector determination: the average orientation of these gradients will be a good approximation of the normal orientation in p. This is not the case if p is close to a sharp edge, or close to an other object: the normal vector orientation between p and its neighboring voxels can change drastically. Here, only the gradient vectors whose orientations are close to that of p are relevant

Lowres version

for the normal estimation. By discarding gradient vectors whose orientations are far from the gradient vector orientation in p, we can construct an adapted shape neighborhood over which the vectors can be summed.

The angular criterion to apply on gradient vectors is directly related to the maximal angle that should be considered as an edge by the algorithm. Let us express this relationship:

Given $\vec{n}(p)$, the outward normal vector in p and q an other voxel of $\mathcal{O}$, we denote by β the angle $\angle(\vec{n}(p), -\vec{\nabla} d_b(q))$. Let α be the aperture angle of an infinite cone $\mathcal{C}(p, \alpha)$ of summit p whose generatrices are tangential to the object's surface.

Let us define a critical angle α_0 so that, for any angle $\alpha < \alpha_0$, the surface is considered as having an edge. Elementary geometrical laws imply that the corresponding critical angle between $\vec{n}(p)$ and $-\vec{\nabla} d_b(q)$ amounts to:

$$\beta_0 = \frac{\pi - \alpha_0}{2} \tag{3}$$

Thus, a local gradient vector whose angle β is wider than $\frac{\pi-\alpha_0}{2}$ has to be ignored in the summation (Fig. 2). Hence, we have the following angular criterion:

Criterion 1 (angular criterion) $\vec{\nabla} d_b(q)$ has to be taken into account if and only if:

$$\frac{-\vec{\nabla} d_b(q).\vec{n}(p)}{|\vec{\nabla} d_b(q)|.|\vec{n}(p)|} < \cos\beta_0 \tag{4}$$

with $\beta_0 = \frac{\pi-\alpha_0}{2}$, α_0 being the critical angle.

The control parameter α_0 is the angular width of the smoothest detail that should be accounted for in the considered problem. Note that α_0 must be acute enough to obtain a smooth vector field and wide enough to detect singularities. In our case, we chose $\alpha_0 = 120°$. This angle provides good results and is particularly relevant for snow images for physical reasons: 120° corresponds to the maximal angle of hexagonal ice crystals. As these shapes are common in our images, we wanted to process them correctly.

Symmetry criterion:

If p belongs to a smooth surface, applying the angular criterion leads to average the gradient vector orientations on a spherical neighborhood around p. On the contrary, if sharp variations are detected in the gradient field, the neighborhood can be asymmetrical with regard to the normal vector $\vec{n}(p)$. In this case, summing the gradient vectors of such a neighborhood can lead to a false normal vector: by taking into account too many selected contributions, this neigborhood may not be representative of the surface in p. This phenomenon is particularly visible when rounded shapes are close to wide flat surfaces (see Fig. 3).

Lowres version

To ensure the validity of the normal vector at point p, the effective neighborhood $\mathcal{V}(p)$ should be restricted to a $\vec{n}(p)$-symmetrical neighborhood (Fig. 2). Hence, we have the following symmetry criterion:

Criterion 2 (symmetry criterion) $\vec{\nabla} d_b(q)$ is taken into account if and only if:

$$sym(q, \vec{n}(p)) \in \mathcal{V}(p) \tag{5}$$

where $sym(q, \vec{n}(p))$ denotes the symmetric voxel of q according to the vector $\vec{n}(p)$.

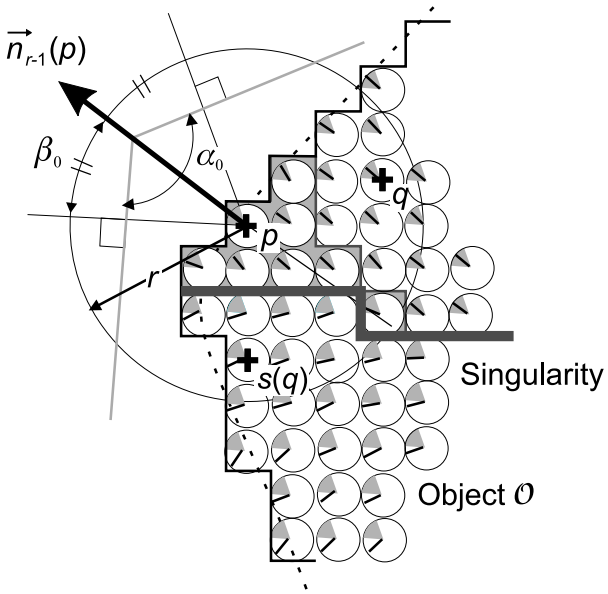


Fig. 2. Summation of gradient vectors in 2D. For each pixel, the gradient vector is represented by a bar in a circle. A gray angular sector, depending on α_0 and $\vec{n}_{r-1}(p)$, is plotted in each pixel. The bar is embedded in this gray sector for the pixels verifying the angular criterion. The pixel area that is finally retained for the determination of $\vec{n}_r(p)$ is filled in gray (angular and symmetry criteria).

Algorithm:

In order to determine $\vec{n}(p)$ on a point $p \in \mathcal{S}$, angular and symmetry criteria are tested on a spherical neighborhood of radius r_{max}. Since these two criteria depend on $\vec{n}(p)$, we chose an iterative approach to obtain the normal vector at p.

Let $\vec{n}_r(p)$ be the estimation of the normal vector using an effective neighborhood included in a r radius sphere.

$-\vec{\nabla} d_b(p)$ is assumed to be the first estimation of the normal vector. $\vec{n}_r(p)$ is updated by summing each relevant contribution of the r-radius spherical cap to the previous normal estimation $\vec{n}_{r-1}(p)$.

Here is the general algorithm:

Lowres version

$\vec{n}(p)_0 = -\vec{\nabla} d_b(p)$

While $(r < r_{max})$

{

$r = r + 1$

$\vec{n}_r(p) = \vec{n}_{r-1}(p) + \sum_{q \in \mathcal{V}_r(p)} [-\vec{\nabla} d_b(p)]$

}

$\vec{n}(p)_{r_{max}}$ is normalized.

$\mathcal{V}_r(p) = \{q \in \mathcal{O}\}$ so that:

$$\begin{cases} d(p,q) = r \\ s(q) \in \mathcal{O} \\ \frac{-\vec{\nabla} d_b(q).\vec{n}_{r-1}(p)}{|\vec{\nabla} d_b(q)|.|\vec{n}_{r-1}(p)|} < \cos\beta_0 \\ \\ \frac{-\vec{\nabla} d_b(s(q)).\vec{n}_{r-1}(p)}{|\vec{\nabla} d_b(s(q))|.|\vec{n}_{r-1}(p)|} < \cos\beta_0 \end{cases}$$

with $s(q) \equiv sym(q, \vec{n}_{r-1}(p))$

Note that if $r_{max} = 0$, the resulting normal vector in p is the original gradient vector.

A.4 Remarks on the ADGF algorithm: possible extensions

Here we propose some extensions that can improve the accuracy of the presented algorithm. The counterpart of this accuracy is an increase of computation time.

Signed distance map:

In the ADGF algorithm, the normal vectors are computed from the inner side of the object. This method can be extended to the whole image by computing a signed distance map [27], [2], positive in the foreground and negative in the background. With such a distance map, the computation of the gradient field can be carried out everywhere in the image. This has several advantages:

- the gradient vectors belonging to $\mathcal{S}$, which are used as the first estimates of the normal vectors, are better estimated.
- taking into account the background gradients in the summation algorithm gives a more robust estimation of the normal field on highly convex shapes.
- higher order derivatives (like mean curvature [2]) can be computed on $\mathcal{S}$.

Fréchet expectation:

In the presented algorithm, each estimation of the normal vector $\vec{n}_r(p)$ is realized by summing

Lowres version

$\vec{n}_{r-1}(p)$ to the unit gradient vectors that belong to $\mathcal{V}_r(p)$. A more rigorous, but computationaly intensive approach can consist in estimating $\vec{n}_r(p)$ with a minimization process like the Fréchet expectation [28], [29].

Connectivity criterion:

To be fully adaptive on all types of images, the algorithm should reject the points of $\mathcal{V}_r(p)$ which are disconnected from the points belonging to $\mathcal{V}_{r-1}(p)$. For our snow images at a resolution of 10 microns, the connectivity test can be easily avoided by choosing a relatively low value of r_{max}. For example $r_{max} = 5$ voxels gives very good results. The choice of r_{max} will be discussed in the experiment part.

B. Validations

The ADGF algorithm was tested on simulated data and compared to Marching Cubes algorithms.

The Marching Cubes algorithms used are the following:

- The simple Marching Cubes algorithm (MC for short) as presented in [4].
- The Smoothed Marching Cubes algorithm (SMC for short): it is obtained by preprocessing the original black and white image in order to smooth the "steps" commonly produced by the MC algorithm. In our case, black and white data were converted into grayscale images by applying a classical 5^3-size mean kernel.
- The Gaussian Smoothed Marching Cubes algorithm (GSMC for short): it is obtained by preprocessing the original black and white image with a 11^3-size Gaussian kernel. The half-width σ of the chosen Gaussian function was set to 2 voxels.

These three MC methods were chosen to point out the ADGF adaptivity. The MC method does not have any smoothing process and is highly sensitive to digitization effects. Nevertheless, it gives accurate results on sharp edges. The GSMC method is weakly sensitive to digitization effects but smoothes fine shapes (Fig. 7). The SMC algorithm is a fairly good compromise between the two other methods.

To allow comparison with the ADGF algorithm, the facets obtained by Marching Cubes algorithms are processed as follows:

For each facet:

- the barycenter of the facet is computed.
- the normal vector is obtained by calculating the half cross product of two vectors generating the facet.
- the normal is assigned to the nearest center of the Marching Cubes grid (which is in fact a

Lowres version

node of the voxel grid).

• the normal is weighted by a distance-depending Gaussian filter and this contribution is added in the eight neighboring voxels of this center.

By summing all the facet contributions, one can obtain the appropriate normal vector in each voxel. Then, for a voxel belonging to the surface of the object, the resulting normal vector is considered as the Marching Cubes normal vector.

The quality evaluation of the normal vectors was realized in two ways:

• by comparing the obtained normal vectors to the theoretical normal vectors for synthetic objects whose surface is differentiable.

• by visualizing them for other objects. For visualization, $\vec{u}$ being the illumination vector, the gray level in p was simply taken as proportional to $-\vec{n}(p).\vec{u}+1$.

Except for the first test where the effect of r_{max} was analyzed, r_{max} was set to 5 voxels.

B.1 Effects of the neighborhood maximal size

For this test, an object whose normal vector field is well-known but relatively complex was chosen. The surface equation of this object: $x^6+y^6+z^6=35^6$ allows to define the theoretical normal vector field on each point of its surface as follows:

$$\vec{n}_{th}(x,y,z) = 6x^5.\vec{i} + 6y^5.\vec{j} + 6z^5.\vec{k} \tag{6}$$

$(\vec{i},\vec{j},\vec{k})$ being the canonical basis of $\mathbb{R}^3$.

We name the objects defined by such a surface "rounded cubes". To obtain a tilted rounded cube, a rotation of the object and of its theoretical vector field was done (Fig. 4(b)). To evidence the effects of the choice of r_{max}, we chose a rotation in order to low tilt the plane parts of the rounded cube with respect to the grid axes. In Fig. 3 are presented the results given by the ADGF algorithm with and without the symmetry criterion. For each method, the mean error and standard deviation are plotted versus r_{max} from 0 to 24 voxels. Rendering images are presented in Fig. 4.

From these figures, it appears clearly that:

• the symmetry criterion is necessary to have an adaptive algorithm on smooth shapes (Fig. 3).

• the normal vector needs to be computed from a sufficient number of gradient vectors. Otherwise, it is corrupted by digitization effects (Fig. 4(c)). Nevertheless, a small neighborhood (4-5 voxels) is sufficient to obtain good results (Fig. 4(d)).

Lowres version

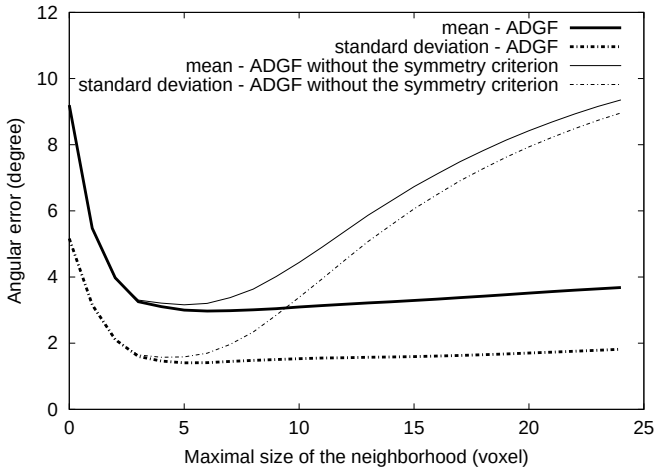
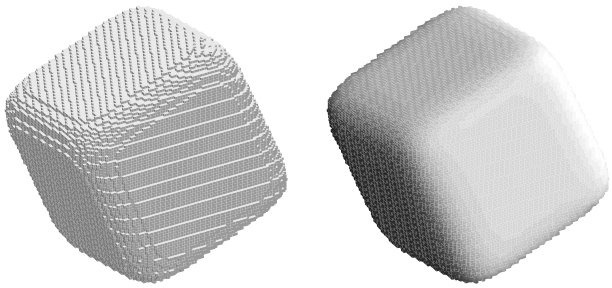
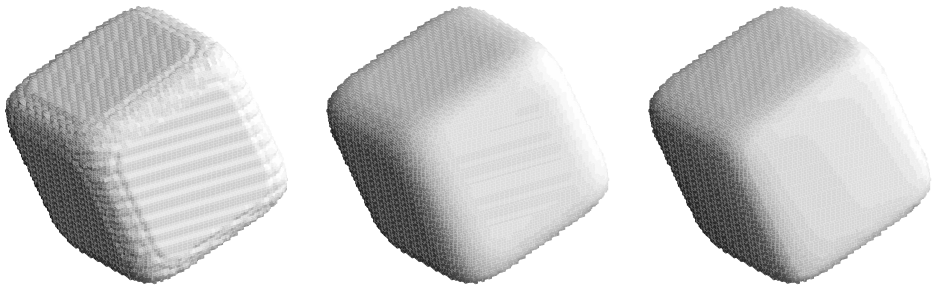


Fig. 3. Angular error variation with the size of the neighborhood for a tilted rounded cube. Note the importance of the symmetry criterion for the ADGF adaptivity. The considered rounded cube was especially chosen to give high errors: generally, the errors are smaller (see Fig. 6(b)).



(a) Voxel image. (b) Theoretical vectors.



(c) $r_{max} = 0$. (d) $r_{max} = 5$. (e) $r_{max} = 24$.

Fig. 4. ADGF algorithm with different effective neighborhood maximal sizes. Image renderings of the surface of equation: $x^6 + y^6 + z^6 = 35^6$.

Lowres version

• the maximal size of the neighborhood has to be close to the size of the smallest detail we want to process correctly. Nevertheless, even if the neighborhood is very large, the error on the normal vector field remains low (Fig. 3 and 4(e)).

These two last remarks point out the adaptivity of the method : by choosing a relatively high value of r_{max}, the algorithm gives very good results on all types of shapes.

B.2 Multigrid tests on spheres

Here is empirically adressed the multigrid convergence of normals (see section II-C). Tests were carried out on a set of digitized spheres from 1 to 80 voxels of radius. Spheres were obtained according to the definition adopted in [11] and [13]: a sphere with an integer radius r is defined by $(x, y, z) \in \mathbb{Z}^3$ so that:

$$x^2 + y^2 + z^2 < r.(r + 1) \tag{7}$$

The theoretical normal vector is the vector starting from the sphere center through the sphere voxel centers. Errors obtained with the ADGF and MC algorithms are compared in Fig. 5. We can note that the ADGF algorithm gives results close to those obtained with the GSMC method, which is specially adapted to big spheres. This proves the convergence of the ADGF algorithm in the continuous limit.

B.3 Sensitivity to the surface orientation

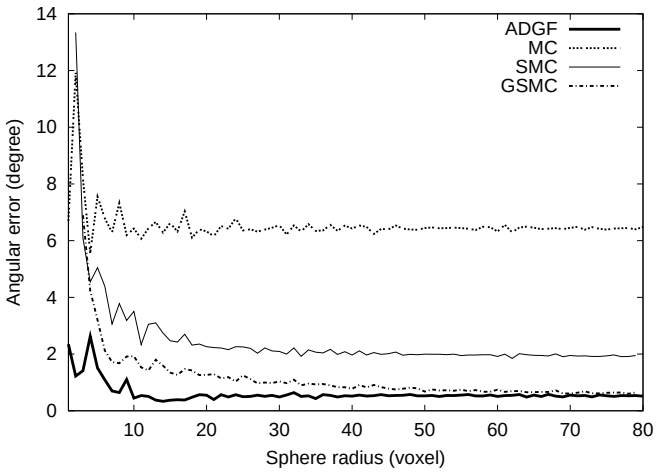
The sensitivity to the surface orientation was firstly estimated on digital planes of various orientations. Planes were generated in thirty evenly spaced different orientations. Then, for each voxel of each plane, the normal vector obtained was compared to the corresponding theoretical normal vector. Counting, for the thirty planes, the number of error values corresponding to a fixed error class leads to the graph of the Fig. 6(a).

The same test, applied on thirty tilted rounded cubes is presented in Fig. 6(b). Tilted rounded cubes and theoretical vectors were obtained as explained above (see section III-B.1).

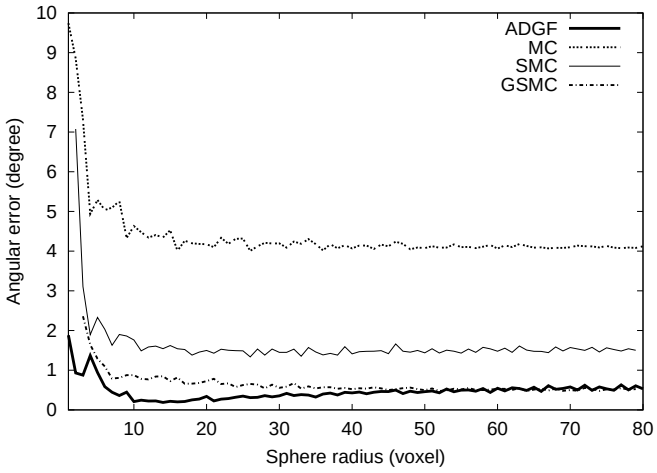
For such tests, the GSMC algorithm, weakly sensitive to digitization effects, is particularly adapted and gives very good results. Note that, for infinite planes, the size of the ADGF neighborhood is always maximal: the ADGF and the GSMC algorithms have then the same neighborhood and this explains the similar histograms observed.

The test with rounded cubes shows the precision and the sensitivity of the ADGF method when its adaptivity is activated. In this case, the ADGF peak is almost as sharp as the GSMC peak: the ADGF algorithm is precise and weakly sensitive to the surface orientation.

Lowres version



(a) Mean angular error



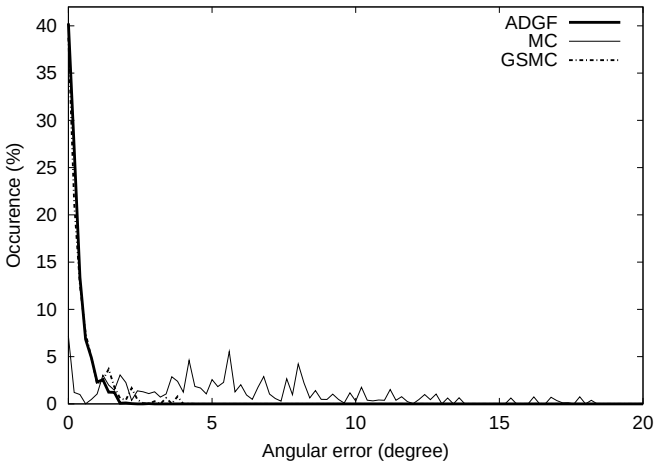
(b) Standard deviation of the angular error

Fig. 5. Comparison of the angular error for multigrid spheres with the different algorithms.

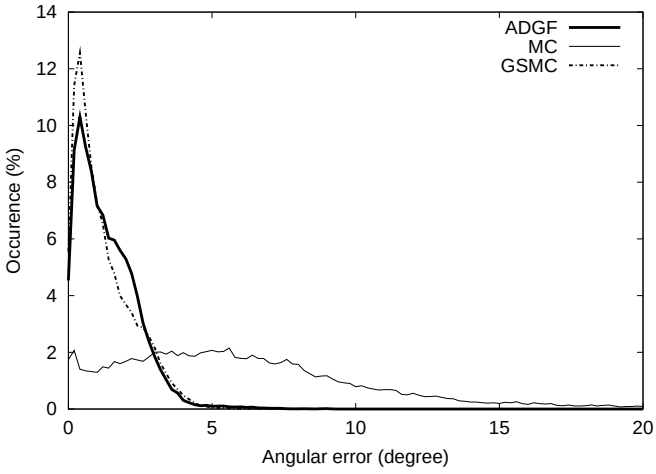
B.4 Other geometrical shapes

Here are compared ADGF and MC algorithms on other simulated data. Rendering images of the Fig. 7 point out the accuracy of ADGF method on convex, concave, flat, rounded and sharp shapes.

Lowres version



(a) Digital planes.



(b) Tilted rounded cubes.

Fig. 6. Sensitivity to the surface orientation: comparison of the angular error for objects taken in thirty different orientations. The histogram was obtained by counting, for the thirty objects, the number of error values corresponding to a fixed error class. Each class is 0.2 degree wide.

Lowres version

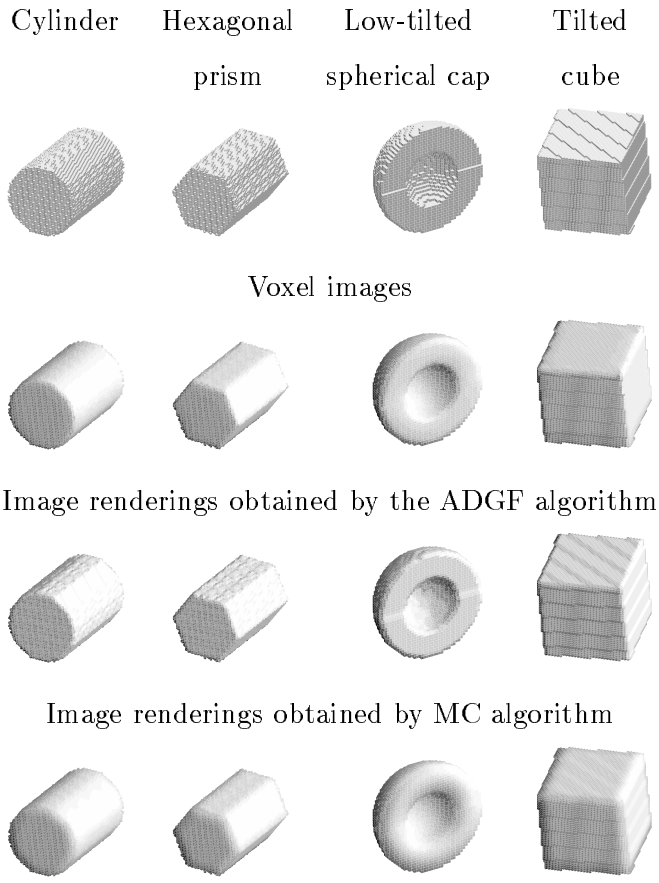


Fig. 7. Comparison between the ADGF and MC algorithms on geometrical shapes.

Lowres version

IV. Surface area estimation

A. From normal vectors to surface area estimation

We have presented a new and simple computational method to obtain accurate normal vectors on the surface of a numerical object. This method is more time-consuming than the Marching Cubes algorithms but allows to compute a high quality normal vector field. Such a vector field can be used for image rendering, local curvature estimation [30] and many other purposes. In particular, it can be used for estimating the surface area of a numerical object with high accuracy. Classical approaches consider a polygonalization of the raw data and estimate the surface area by summing the area of the facets. It is the case of the Convex Hull or the Marching Cubes algorithms. In this section, we present a surface area estimator directly based on the discrete model.

A.1 Statement of the problem

We consider the field of normal vectors $\mathcal{N} = \{\vec{n}_i\}$ defined at each voxel of the discrete surface $dig_r(\mathcal{S})$. The surface area of a discrete object can be obtained within the limits of the tangent plane approximation, as a sum of the area contributions of each surface voxel. When a surfel decomposition of the object is available, projecting all surfels of each voxel along $\vec{n}$ directly provides the contributions [16], [12]. Such a projection along $\vec{n}$ cannot be simply extended for voxel data: as the number of surfels per voxel depends on the local configuration [31], [32], obtaining the surfel decomposition of the object is needed.

Here, we describe a method to obtain directly the surface area from raw voxels, without doing the surfel decomposition.

A.2 Voxel projection algorithm

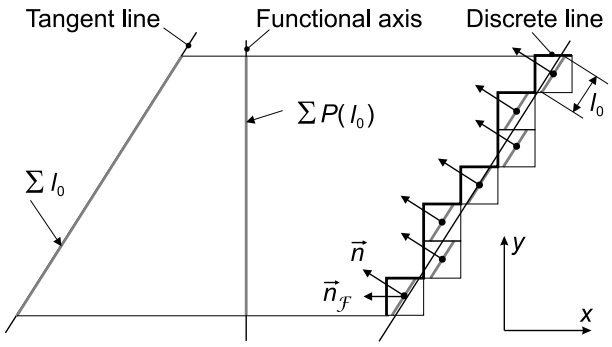
As the method of Lenoir [16], [12], this method is based on projections and assume the tangent plane approximation. But rather than using a systematic surfel projection along $\vec{n}$, we consider, for each surface voxel, a projection of the normal plane on an appropriate surfel.

Let $\mathcal{P}$ be a discrete plane and Π the associated continuous plane so that Π is orthogonal to $\vec{n}$. Let us define $\mathcal{F}$, the coordinate plane such that $\mathcal{P}$ is functional to $\mathcal{F}$ [33] (*i.e.* there is a one to one map between voxels belonging to $\mathcal{P}$ and their projection on $\mathcal{F}$).

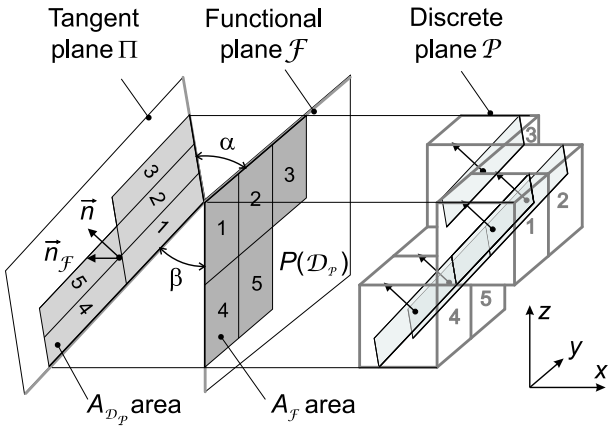
The projection $P(D_\mathcal{P})$ of a domain $D_\mathcal{P}$ of $\mathcal{P}$ on $\mathcal{F}$ has the same area (denoted $A_\mathcal{F}$ in the following) as the sum of surfels of $D_\mathcal{P}$ that are parallel to $\mathcal{F}$ (see Fig. 8).

Conversely, back-projecting this set of surfels and $P(D_\mathcal{P})$ onto Π perpendicularly to $\mathcal{F}$ preserves the equality of areas: the area $A_{D_\mathcal{P}}$ of the domain $D_\mathcal{P}$ is equal to the sum of the

Lowres version



(a) In 2D.



(b) In 3D.

Fig. 8. Illustration of the voxel projections.

back-projected areas of functional surfels belonging to $D_{\mathcal{P}}$. By definition, the projection ratio $ratio(\vec{n})$ of such a projection is:

$$ratio(\vec{n}) = \frac{A_{D_{\mathcal{P}}}(\vec{n})}{A_{\mathcal{F}}} \tag{8}$$

Let n_x, n_y, n_z denote the coordinates of $\vec{n}$, $\vec{X}_{\mathcal{F}}$ the outward unit normal of the functional surfel s and $\vec{n}_{\mathcal{F}}$ the orthogonal projection of $\vec{n}$ on $\vec{X}_{\mathcal{F}}$. Then, the above ratio can be expressed as follows:

$$ratio(\vec{n}) = \frac{||\vec{n}||}{||\vec{n}_{\mathcal{F}}||} \tag{9}$$

By construction, we have: $||\vec{n}_{\mathcal{F}}|| = max(|n_x|, |n_y|, |n_z|)$. Hence,

$$ratio(\vec{n}) = \frac{||\vec{n}||}{max(|n_x|, |n_y|, |n_z|)} \tag{10}$$

The surface area contribution of a surface voxel is given by multiplying this ratio by the area of a surfel. This ratio only depends on $\vec{n}$: it is evaluated for each surface voxel of normal vector $\vec{n}$.

Lowres version

The sum of each voxel contribution provides the area of the whole object in surfel units. Thus, we obtain the following surface area estimator:

$$E_{\mathcal{A}}(dig_r(\mathcal{S})) = \sum_{\vec{n}_i \in \mathcal{N}} ratio(\vec{n}_i) \tag{11}$$

B. Validations

The voxel projection estimator (VP for short) was tested on simulated data and compared to other estimators. The quality evaluation of the surface area estimator was realized in two ways:

- by comparing the convergence of the voxel projection estimator with other surface area estimators (Convex Hull[1] [34] and MC methods) for objects whose surface area is known theoretically.
- by visualizing the area ratios calculated in each voxel of simple objects.

B.1 Convergence tests

Here is empirically adressed the multigrid convergence of surface area estimations (see section II-C). The convergence tests were carried out on spheres, cylinders and thick spherical caps. Relative errors of surface area estimation are plotted against grid resolution.
The series of spheres are the same as defined in the first part of this article.
The cylinders verify the following conditions:

- the height amounts to twice the radius of the disk plus one voxel.
- the cylinder axis is colinear to the (Oz) axis.

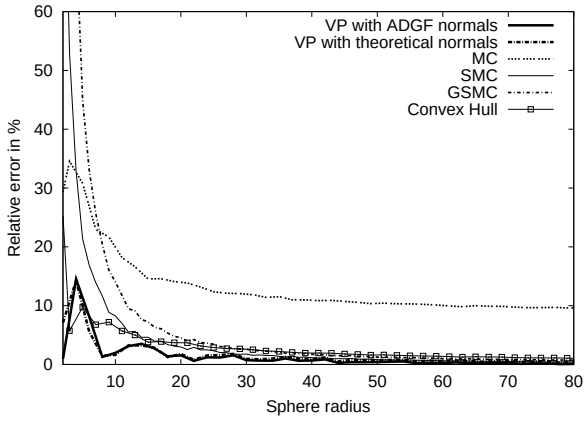
Thick spherical caps were obtained as follows:

- the outer sphere radius is twice the inner sphere radius.
- the two spheres verify the equation 7.
- to make an easier theoretical estimation of this object, the two spheres were truncated so that their surface area is the half part of a whole sphere surface area (see the voxel image of the low-tilted spherical cap in Fig. 7).

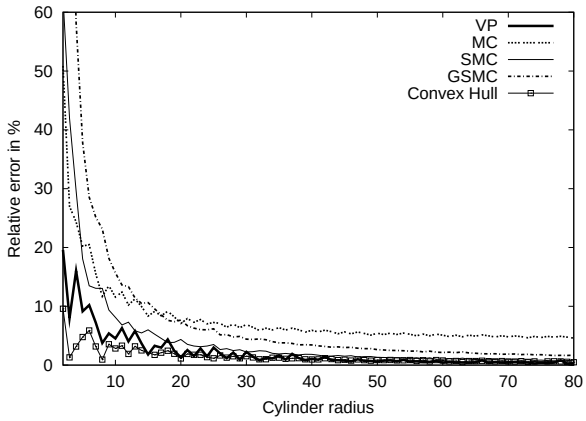
The convergence rates of both Convex Hull and VP surface area estimators are better than the convergence rates of the tested MC algorithms (Fig. 9). In addition, unlike the Convex Hull method, the VP estimator allows to compute surface areas of non-convex objects. In Fig. 9(a), the surface area estimator does not suffer from the ADGF normal vector estimation: curves with theoretical normal vectors and ADGF-estimated vectors are similar.

[1]Convex Hull estimations were obtained with "Qhull". http://www.geom.uiuc.edu/software/download/qhull.html

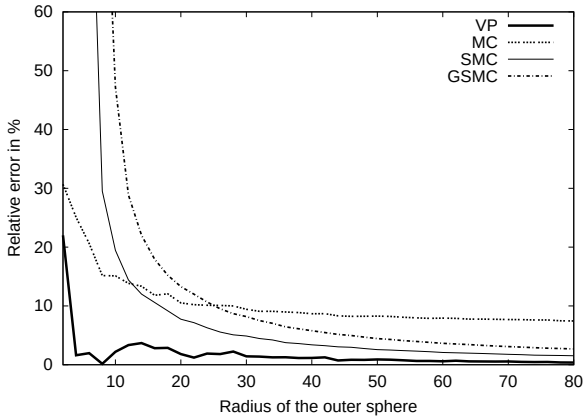
Lowres version



(a) Spheres.



(b) Cylinders.



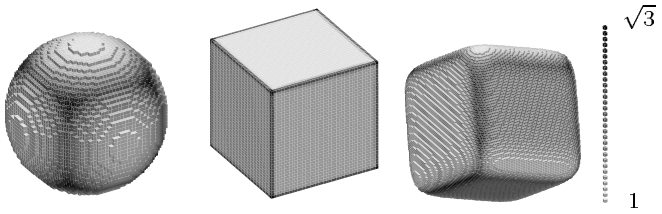
(c) Thick spherical caps.

Fig. 9. Comparisons of the surface area estimation for multigrid objects using different algorithms.

Lowres version

B.2 Local surface area ratio visualization

Local surface area ratio maps were obtained as follows: the ADGF normal determination algorithm was applied to geometrical objects. Then the projection ratios were computed and plotted in gray level for each point of the discrete surface (Fig. 10).



(a) (b) (c)

Fig. 10. Maps of surface area ratios estimated on geometric objects: sphere (a), cube "in axes" (b) and tilted rounded cube (c). Surface area ratios are ranged from 1 to $\sqrt{3}$.

V. Application on snow samples: ADGF versus MC

A. Estimation of the normals

Here are presented some image renderings of snow samples obtained by microtomography. These samples correspond to different snow types at different resolutions: rounded grains (Fig. 11), fresh snow (Fig. 12) and melt-freeze crust (Fig. 13). These types of snow are constituted of both rounded shapes and sharp angles and allow to check the effectiveness of the ADGF algorithm by comparison with Marching Cubes methods. For fresh snow and melt-freeze crust, the ADGF renderings are compared with image renderings obtained with the Marching Cubes methods. The ADGF results are particularly impressive for fresh snow, where some plates are only 3-4 voxel thick.

B. Surface area estimation

The surface area of three snow samples was computed by Marching Cubes and ADGF methods. The comparison of surface area estimations of snow sample images are presented in table I. For these estimations, the whole numerical object was taken into account.

The three surface area estimators give relatively similar results, except for the GSMC method on fresh snow: in this case, significant parts of the object are not properly processed due to the size of the kernel used (see Fig. 12(c)). We can see from Fig. 13(b) that the roughness of the surface described by the MC algorithm is high. Thus, the surface area using this method

Lowres version

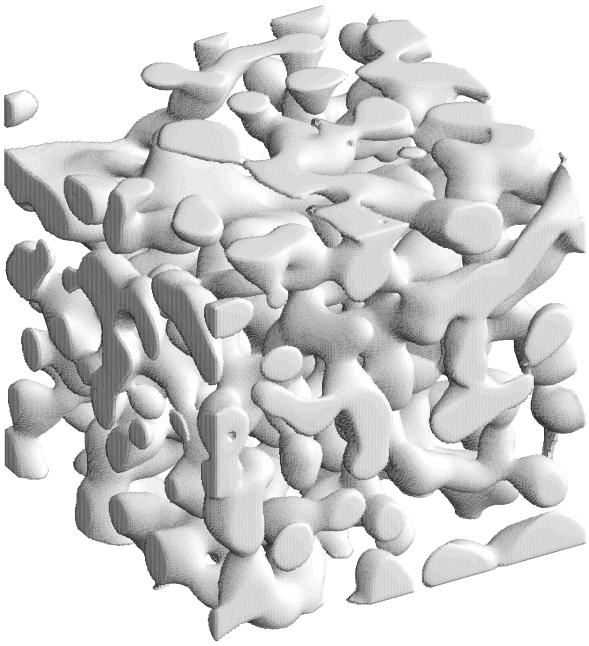
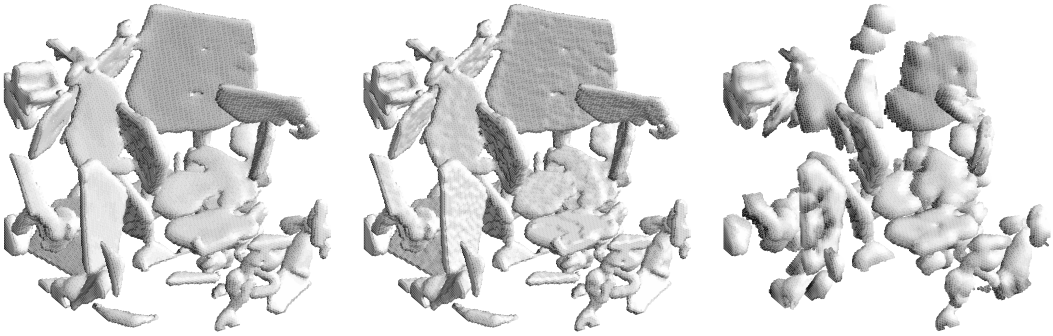


Fig. 11. Image renderings obtained by the ADGF algorithm on a sample of rounded grains. Image frame: 256^3 voxels.



(a) ADGF algorithm. (b) MC algorithm. (c) GSMC algorithm.

Fig. 12. Image renderings of a fresh snow sample with the three methods. Image frame: 128^3 voxels. The MC algorithm (b) is highly sensitive to digitization effects. The GSMC (c) algorithm smoothes the digitization steps but deletes significant data: fine snow plates are not properly processed due to the size of the gaussian kernel used. Only the ADGF algorithm (a) preserves small details while smoothing the digitization artefacts.

Lowres version

(a) ADGF algorithm. (b) MC algorithm. (c) GSMC algorithm.

Fig. 13. Image renderings of a snow sample (melt-freeze crust taken at le Col de Porte, France) with the three methods (detail). Frame of the original image: 301^3 voxels. The MC algorithm (b) is highly sensitive to digitization effects. The GSMC algorithm (c) smoothes the digitization steps but deletes relevant details (circle). Only the ADGF algorithm (a) preserves small details while smoothing the digitization artefacts.

TABLE I

SURFACE AREAS ESTIMATED ON THREE SNOW SAMPLES (FRESH SNOW, ROUNDED GRAINS AND MELT-FREEZE CRUST) BY THE THREE ALGORITHMS. THE VP ALGORITHM WAS APPLIED TO THE NORMALS OBTAINED BY THE ADGF METHOD. THE DEVIATION (DEV.) BETWEEN THE MARCHING CUBES ESTIMATIONS AND THE VP ALGORITHM IS EXPRESSED IN PERCENT OF THE VP ESTIMATION.

Snow volume		Fresh snow	Rounded grains	Melt-freeze crust
Edge Size (voxel)		128	256	301
MC	area (pixel)	107 998	923 001	1 069 650
	dev. (%)	+15	+9.9	+7.2
GSMC	area (pixel)	33 845	748 517	931 917
	dev. (%)	−64	−10.9	−6.5
VP	area (pixel)	93 921	840 114	997 563

Lowres version

is overestimated. The roughness of the surface processed by the GSMC algorithm (Fig. 13(c)) is much lower. However the surface area of sharp edges, due to edge smoothing, is generally underestimated with this method. Note that the voxel projection estimator gives a surface area value which is included between the two Marching Cubes estimations. This proves the adaptivity of the ADGF algorithm on arbitrary shapes.

VI. Conclusion

A new numerical method to evaluate the field of normal vectors from binary voxel images was presented. It uses volume information generated from the object's surface to reduce digitization effects. The gradient field of the discrete distance, which approximates the normal field, is adaptively filtered around each surface voxel using angle and symmetry criteria, so that as many relevant contributions as possible are accounted for. This allows to optimize the accuracy of both sharp and smooth regions on the same image. Given this precise normal field, a projection method was then proposed to derive immediately the surface area from a raw discrete object. Both techniques were tested on smooth and sharp geometrical shapes, then applied to natural snow tomographic data and compared to Marching Cubes results.

These successful numerical tests will be enforced by a more physical validation: a joint experiment with glaciologists (LGGE, Grenoble) is planned to compare, on a same snow sample, surface area values from the above presented method, and methane adsorption measurements at 77K [35].

The two algorithms presented in this article offer interesting outlooks for modeling snow metamorphism. Since the surface area is a "bulk" geometrical property of a given type of snow, a numerical measurement of surface area is expected to allow a simpler parametrization of the thermodynamics of microstructures in large-scale snow cover models [36]. At the grain-scale, a model of curvature-driven snow metamorphism (very low temperature gradient) has been implemented [3]. The preliminary results [37] are in good agreement with real data from an experiment of isothermal metamorphism hold in a cold room [2].

Lowres version

References

[1] S. C. Colbeck, "A review of sintering in seasonal snow," Tech. Rep. 97-10, U.S. Army Cold Regions Research and Engineering Laboratory, Dec. 1997.

[2] F. Flin, J.-B. Brzoska, B. Lesaffre, and C. Coléou an d R. A. Pieritz, "3D geometric measurements of snow microstructural evolution under isothermal conditions," *Ann. Glaciol.*, vol. 38, 2004, in press.

[3] F. Flin, *Description physique des métamorphoses de la neige à partir d'images de microstructures 3D naturelles obtenues par microtomographie X*, Ph.D. thesis, Université Joseph Fourier, Grenoble, 2004, in French.

[4] W. E. Lorensen and H. E. Cline, "Marching cubes: a high resolution 3D surface construction algorithm," *Computer Graphics*, vol. 21, no. 4, pp. 163–169, 1987, SIGGRAPH'87 Conference Proceedings (Anaheim, California).

[5] H. Gouraud, "Continuous shading of surfaces," *IEEE Trans. Comput.*, vol. C-20, no. 6, pp. 223–228, Jun. 1971.

[6] P.-L. George and H. Borouchaki, *Delaunay triangulation and meshing - Application to finite elements*, Hermes, Paris, 1998, in French.

[7] R. Yagel, D. Cohen, and A. Kaufman, "Normal estimation in 3D discrete space," *The visual computer*, vol. 8, no. 5-6, pp. 278–291, Jun. 1992.

[8] G. Thürmer and C. A. Wüthrich, "Normal computation for discrete surfaces in 3D space," *Computer Graphics Forum*, vol. 16, no. 3, pp. 15–26, Aug. 1997, Proceedings of Eurographics'97.

[9] V. A. Kovalevsky, "Finite topology as applied to image analysis," *Comput. Vision Graphics Image Process.*, vol. 46, no. 2, pp. 141–161, May 1989.

[10] L. Chen, G. T. Herman, R. A. Reynolds, and J. K. Udupa, "Surface shading in the Cuberille environment," *IEEE Comput. Graph. Appl.*, vol. 5, no. 12, pp. 33–43, Dec. 1985.

[11] L. Papier and J. Françon, "Evaluation de la normale au bord d'un objet discret 3D," *Revue de CFAO et d'informatique graphique*, vol. 13, pp. 205–226, 1998, in French.

[12] A. Lenoir, R. Malgouyres, and M. Revenu, "Fast computation of the normal vector field of the surface of a 3-D discrete object," in *Discrete Geometry for Computer Imagery: 6th Workshop*, S. Miguet, A. Montanvert, and S. Ubéda, Eds. 1996, vol. 1176 of *Lect. Notes Comput. Sci.*, pp. 101–112, Springer-Verlag.

[13] P. Tellier and I. Debled-Renesson, "3D discrete normal vectors," in *Discrete Geometry for Computer Imagery: 8th Workshop*, G. Bertrand, M. Couprie, and L. Perroton, Eds. 1999, vol. 1568 of *Lect. Notes Comput. Sci.*, pp. 447–458, Springer-Verlag.

[14] G. Thürmer and C. A. Wüthrich, "Varying neighborhood parameters for the computation of normals on surfaces in discrete space," in *Proceedings of the Conference on Computer Graphics International 1998 (CGI-98)*, F.-E. Wolter and N. M. Patrikalakis, Eds. 1998, pp. 616–625, IEEE Computer Society.

[15] F. Flin, J.-B. Brzoska, B. Lesaffre, C. Coléou, and P. Lamboley, "Computation of normal vectors of discrete 3D objects: application to natural snow images from X-ray tomography," *Image Anal. Stereol.*, vol. 20, pp. 187–191, Nov. 2001.

[16] A. Lenoir, *Des outils pour les surfaces discrètes*, Ph.D. thesis, Université de Caen, 1999, in French, http://www.greyc.ismra.fr/EquipeImage/Pub/PostScript/tlenoir.ps.gz.

[17] T. Y. Kong and A. Rosenfeld, "Digital topology. Introduction and survey," *Comput. Vision Graphics Image Process.*, vol. 48, no. 3, pp. 357–393, Dec. 1989.

Lowres version

[18] D. G. Morgenthaler and A. Rosenfeld, "Surfaces in three-dimensional digital images," *Information and Control*, vol. 51, no. 3, pp. 227–247, Dec. 1981.

[19] P. E. Danielsson, "Euclidean distance mapping," *Comput. Graphics Image Process.*, vol. 14, no. 3, pp. 227–248, Nov. 1980.

[20] I. Ragnemalm, "The euclidean distance transform in arbitrary dimensions," *Pattern Recognit. Lett.*, vol. 14, pp. 883–888, 1993.

[21] G. Borgefors, "Distance transformations in digital images," *Comput. Vision Graphics Image Process.*, vol. 34, no. 3, pp. 344–371, Jun. 1986.

[22] J.-M. Chassery and A. Montanvert, *Géométrie discrète en analyse d'images*, Hermes, Paris, 1991, in French.

[23] A. Jonas and N. Kiryati, "Digital representation schemes for 3D curves," *Pattern Recognit.*, vol. 30, no. 11, pp. 1803–1816, 1997.

[24] Y. Kenmochi and R. Klette, "Surface area estimation for digitized regular solids," in *Proc. Vision Geometry IX*, L. J. Latecki, D. M. Mount, and A. Y. Wu, Eds. Oct. 2000, vol. 4117, pp. 100–111, SPIE.

[25] B. J. H. Verwer, "Local distances for distance transformations in two and three dimensions," *Pattern Recognit. Lett.*, vol. 12, pp. 671–682, Nov. 1991.

[26] G. Borgefors and S. Svensson, "Optimal local distances for distance transforms in 3D using an extended neighbourhood," in *International Workshop on Visual Form 4*, C. Arcelli, L.P. Cordella, and G. Sanniti di Baja, Eds. 2001, vol. 2059 of *Lect. Notes Comput. Sci.*, pp. 113–122, Springer-Verlag.

[27] G. T. Herman, J. Zheng, and C. A. Bucholtz, "Shape-based interpolation," *IEEE Comput. Graph. Appl.*, pp. 69–79, May 1992.

[28] X. Pennec and N. Ayache, "Uniform distribution, distance and expectation problems for geometric features processing," *Journal of Mathematical Imaging and Vision*, vol. 9, pp. 49–67, Jul. 1998.

[29] P. J. Rousseeuw and A. M. Leroy, *Robust regression and outlier detection*, J. Wiley and sons, New York, 1987.

[30] J.-B. Brzoska, B. Lesaffre, C. Coléou, K. Xu, and R. A. Pieritz, "Computation of 3D curvatures on a wet snow sample," *Eur. Phys. J. AP*, vol. 7, pp. 45–57, 1999.

[31] J. Vittone, *Caractérisation et reconnaissance de droites et de plans en géométrie discrète*, Ph.D. thesis, Université Joseph Fourier, Grenoble, 1999, in French, http://www.lis.inpg.fr/theses/manuscript/vittone.ps.gz.

[32] J. Vittone and J.-M. Chassery, "Coexistence of tricubes in digital naive plane," in *Discrete Geometry for Computer Science, DGCI'97*, E. Ahronovitz and C. Fioro, Eds. 1997, vol. 1347 of *Lect. Notes Comput. Sci.*, pp. 99–112, Springer-Verlag.

[33] E. Andrès, *Cercles discrets et rotations discrètes*, Ph.D. thesis, Université Louis Pasteur, Strasbourg, 1992, in French.

[34] C. B. Barber, D. P. Dobkin, and H. T. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Trans. on Mathematical Software*, vol. 22, no. 4, pp. 469–483, Dec. 1996.

[35] L. Legagneux, A. Cabanes, and F. Dominé, "Measurement of the specific surface area of 176 snow samples using methane adsorption at 77K," *J. Geophys. Res.*, vol. 107, no. D17, pp. 3335, 2002.

[36] E. Brun, P. David, M. Sudul, and G. Brunot, "A numerical model to simulate snow-cover stratigraphy for operational avalanche forecasting," *J. Glaciol.*, vol. 38, no. 128, pp. 13–22, 1992.

[37] F. Flin, J.-B. Brzoska, B. Lesaffre, C. Coléou, and R. A. Pieritz, "Full three-dimensional modelling of

Lowres version

curvature-dependent snow metamorphism: first results and comparison with experimental tomographic data," *J. Phys. D: Appl. Phys.*, vol. 36, pp. A49–A54, May 2003.

Lowres version

List of Figures

Lowres version

List of Tables

Lowres version