Convolutions on digital surfaces: on the way iterated convolutions behave and preliminary results about curvature estimation

Sébastien Fourey, Rémy Malgouyres

Convolutions on digital surfaces: on the way iterated convolutions behave and preliminary results about curvature estimation

Sébastien Fourey[1] and Rémy Malgoures[2]

[1] GREYC, UMR6072 – ENSICAEN,
6 bd maréchal Juin 14050 Caen CEDEX, France
Sebastien.Fourey@greyc.ensicaen.fr

[2] Univ. Clermont 1, LAIC, EA2146,
BP 86, 63172 Aubière CEDEX, France
Remy.Malgouyres@laic.u-clermont1.fr

This work was supported by the French National Agency of Research under contract GEODIB ANR-06-BLAN-0225.

Abstract

In [2], the authors present a generalized convolution operator for functions defined on digital surfaces. We provide here some extra material related to this notion. Some about the relative isotropy of the way a convolution kernel (or mask) grows when the convolution operator is iterated. We also provide preliminary results about a way to estimate curvatures on a digital surface, using the same convolution operator.

Keywords: Digital surfaces, normal estimation, convolution, curvature.

1 Introduction

In [2], the authors present a method called *on-surface convolution* which extends the classical notion of a 2D digital filter to the case of digital surfaces. We also defined an averaging mask with local support which, when applied with the iterated convolution operator, behaves like an averaging with large support. The interesting property of the latter averaging is the way the resulting weights are distributed: given a digital surface obtained by discretization of a differentiable surface of $\mathbb{R}^3$, the masks isocurves are close to the Riemannian isodistance curves from the center of the mask.

In the same paper, we used the iterated averaging followed by convolutions with differentiation masks to estimate partial derivatives and then normal vectors over a surface. The number of iterations required to achieve a good estimate was determined experimentally on digitized spheres and tori.

We provide here some extra material related to this notion. Some about the above mentioned relative isotropy of the way a convolution kernel (or mask) grows when the convolution operator is iterated. We also provide preliminary results about a way to estimate curvatures on a digital surface, using the same convolution operator.

2 Digital object and digital surfaces

We briefly recall here the notion of a digital surface (following the *cuberille* model).

A *digital object* is a subset of $\mathbb{Z}^3$, the classical three dimensional grid. Such an object is seen as a set of unit cubes called *object voxels* centered at points with integer coordinates. *Background voxels* are voxels that do not belong to the object.

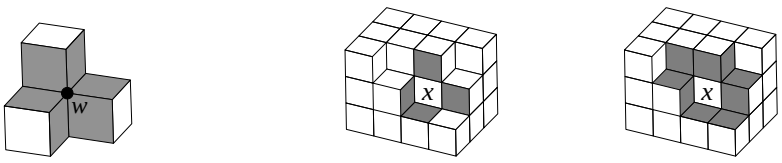
The surface of a digital object is a set of *surfels*, provided with relevant adjacency graphs. Surfels are unit squares that are shared by two 6-adjacent voxels. There are exactly six types of surfels according to the direction of their outward normal vectors. Thus, a surfel is uniquely defined by the data of its center's coordinates and its orientation. In the sequel, a surfel is a pair $(p, \vec{n})$ where $p \in \mathbb{R}^3$ (the center) and $\vec{n} \in \{(\pm 1, 0, 0), (0, \pm 1, 0), (0, 0, \pm 1)\}$ (the normal vector). A *digital surface* is a set of surfels which is the set of all the surfels of a digital object.

We will use in the sequel the two functions σ and ν which associate to a surfel $s = (p, \vec{n})$, respectively, its center $\sigma(s) = p$ and its normal vector $\nu(s) = \vec{n}$.

We can define two adjacency relations between surfels: the e-adjacency and the v-adjacency relations. See [5] for further details.

A relation of e-adjacency (see Figure 1(b)) and v-adjacency (Figure 1(c)) can be defined between some surfels that share an edge or a vertex (see also [3]). In this way, a surfel has exactly 4 e-neighbors, but has a variable number of v-neighbors.

Next, we define a *loop* in a digital surface Σ as an e-connected component of the set of the surfels of Σ which share a given vertex w. For example, if Σ is the surface of the object depicted in Figure 1(a) (which is made of three voxels), then the vertex w defines two loops: one that contains the six gray surfels, and another one in the back with three surfels. Two surfels are *v-adjacent* iff they belong to a common loop of Σ.



(a) A loop of surfels (in gray)

(b) The e-neighborhood of x

(c) The v-neighborhood of x

Figure 1: Loops and neighborhoods on a digital surface.

3 The convolution operator

Here, we recall the definition of the convolution operator given in [1].

In the sequel of this report, Σ is a digital surface and S is a vector space over $\mathbb{R}$. We define the space of *digital surface filters over* Σ as the set of functions from $\Sigma \times \Sigma$ to $\mathbb{R}$.

Definition 1 (Generalized convolution operator) *For $f : \Sigma \longrightarrow S$ and $F : \Sigma \times \Sigma \longrightarrow \mathbb{R}$, we define the operator Ψ as follows:*

$$\begin{array}{rccc} \Psi_{f,F} : & \Sigma & \longrightarrow & S \\ & x & \longmapsto & \displaystyle\sum_{y \in \Sigma} F(x,y) \cdot f(y) \end{array}$$

The operator Ψ acts like a convolution of the values of f on the surface with a convolution kernel whose values should depend on the relative positions of two surfels. We also define the iterated operator $\Psi^{(n)}$.

Definition 2 (Iterated convolution operator) *The* iterated convolution operator *is defined for $n \in \mathbb{N}$ by:*

$$\begin{cases} \Psi^{(0)}_{f,F} = f \\ \Psi^{(n)}_{f,F} = \Psi_{\Psi^{(n-1)}_{f,F},F} \quad \textit{if } n > 0. \end{cases}$$

Now, we may define an averaging kernel. We will later study the isotropy property of the iterated convolution operator with this kernel.

3.1 The averaging filter

We define a local averaging mask $W_{\text{avg}} : \Sigma \times \Sigma \mapsto \mathbb{R}$. This mask should be seen as a wrapping of the 2D classical mask (Figure 2(a)) which follows the *local* shape of the digital surface. The choice of this mask is a heuristic. We tried several masks but this one appears to give particularly good results relating to the Riemannian metrics (see Section 4). Intuitively, we define this mask as a generalization of the $2D$ local mask (and indeed they coincide on a planar surface). The weights are the same as in $2D$ for the e-neighbors, but for the strict v-neighbors (i.e. v-neighbors which are not e-neighbors) the weight of which would be a unique pixel in $2D$ is split and distributed over the several strict v-neighbors of the loop. The global mass of the mask remains unchanged.

More precisely, let x and y be two surfels of Σ such that $y \in N_v(x)$. If y is v-adjacent but not e-adjacent to x then there is a single loop L of Σ that contains both x and y. In this case, we define $\delta_x(y) = \text{card}(L) - 3$. The number $\delta_x(y)$ is used to take into account the number of surfels in a loop containing x which are not e-adjacent or equal to x. Within a loop, all these surfels will end up with a total contribution of $\frac{1}{16}$.

If there are no such surfels, the weight $\frac{1}{16}$ is spread among the two e-neighbors of x in the loop. Thus, if y is e-adjacent to x, then y has exactly two e-neighbors in $N_v(x)$, say s and t. We define $\gamma_x(y)$ as the number of surfels in $\{s,t\}$ which are e-adjacent to x.

Now, let x be a surfel of Σ. For any surfel $y \in \Sigma$ we define the *weight* $W_{\text{avg}}(x,y)$ as follows:

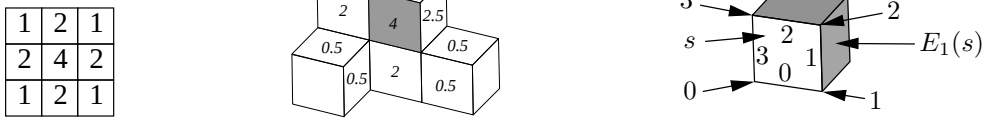
$$W_{\text{avg}}(x,y) = \begin{cases} \frac{1}{4} & \text{if } y = x, \\ \frac{1}{8} + \frac{\gamma_x(y)}{32} & \text{if } y \in N_e(x), \\ \frac{1}{16 \cdot \delta_x(y)} & \text{if } y \in N_v(x) \setminus N_e(x), \\ 0 & \text{if } y \notin N_v(x). \end{cases}$$

We may say that W_{avg} defines an averaging mask because of the following property.

Property 1 ([2]) *For all surfel x of a digital surface Σ we have:*

$$\sum_{y \in \Sigma} W_{avg}(x, y) = 1$$

See Figure 2(b) for an example of a v-neighborhood and the associated values of W_{avg}.



(a) A 2D mask.

(b) A surfel x (in gray) and the values of $16 \cdot W_{\text{avg}}(x, y)$ for the surfels y of $N_v(x) \cup \{x\}$.

(c) Ordering of vertices and edges for a given surfel s.

Figure 2: Illustrations of the masks definition.

4 On the isotropy of the growing mask

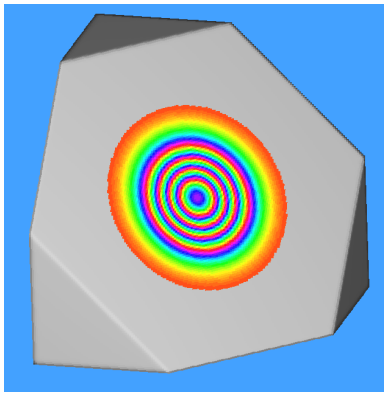
In [2] we claimed and illutrated the fact that the convolution with an averaging mask, when iterated, comes down to a convolution with a large mask that grows according to the number of iteration following an approached Riemannian distance on the digital surface. Futhermore, weights in the obtained larger mask tend to follow the geodesic isodistance within the surface.

In order to illustrate how the averaging mask grows, we use a diffusion process: with $S = \mathbb{R}$ we choose a surfel $s_0 \in \Sigma$ and define the function $\delta_{s_0} : \Sigma \longrightarrow \mathbb{R}$ such that $\delta_{s_0}(s_0) = 10^{30}$ and $\delta_{s_0}(s) = 0$ for $s \in \Sigma \setminus \{s_0\}$. Then, we compute $\Psi^{(n)}_{\delta_{s_0}, W_{\text{avg}}}$ for a given n. Results of this diffusion process that we call an *impulse response of the averaging filter* $\Psi^{(n)}_{f, W_{\text{avg}}}$ are depicted in Figures 3 (from [2]), 4(a) and 4(c).

In Figure 5 we illustrate the deviation between the isocurves of values in the impulse responses and the isolines of the euclidean distance on a digitized plane.

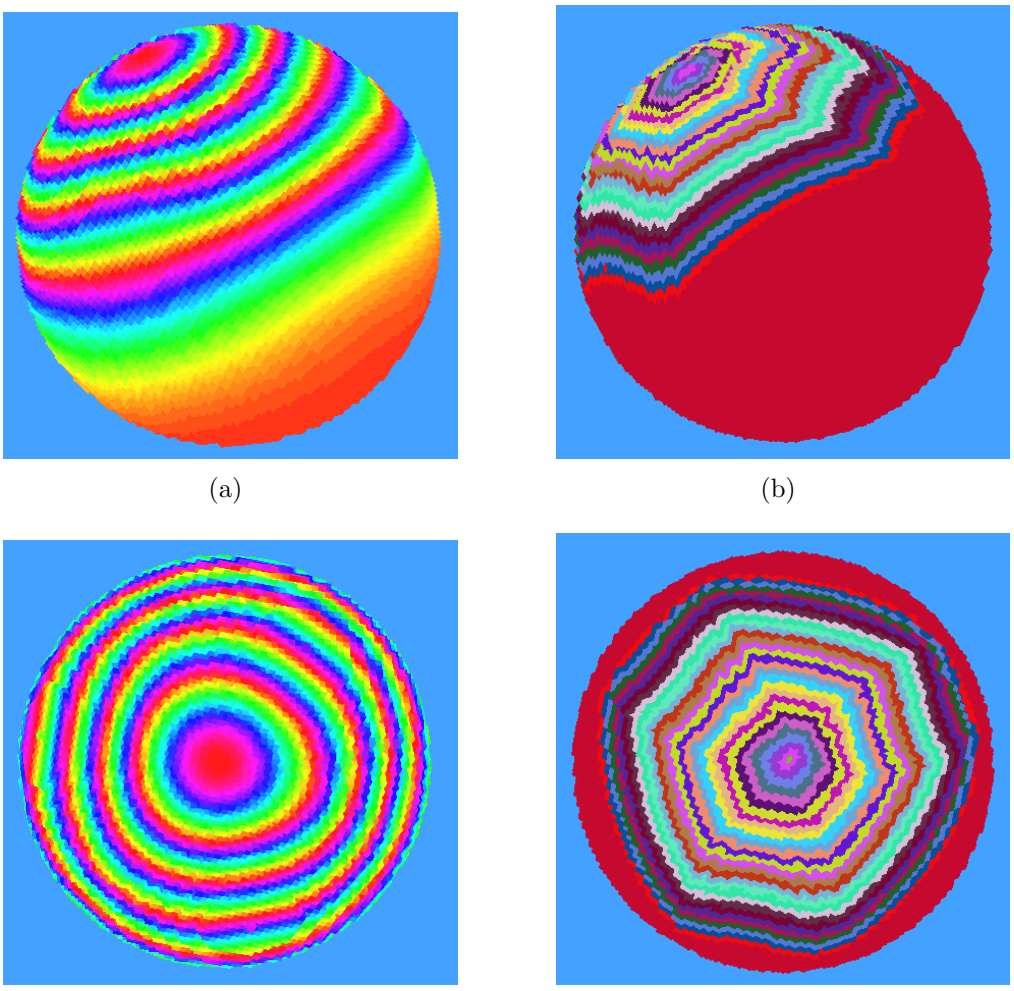
5 Preliminary results on curvature estimation

In this section, we present an attempt to estimate second order quantities after a similar averaging process as the one used to estimate normals. Although the theoretical work done in [4] shows that higher orders estimates may be computed in a similar way for 1D digitized functions, it seems from our first experiments that the precision is not so good when using on-surface convolution.



(a) Plane with $x + y + z = 0$.

Figure 3: Impulse response on a digital plane.



(a) (b) (c) (d)

Figure 4: Impulse responses on a sphere with radius 30 (a,c) compared with the isocurves of the v-adjacency graph distance (b,d).

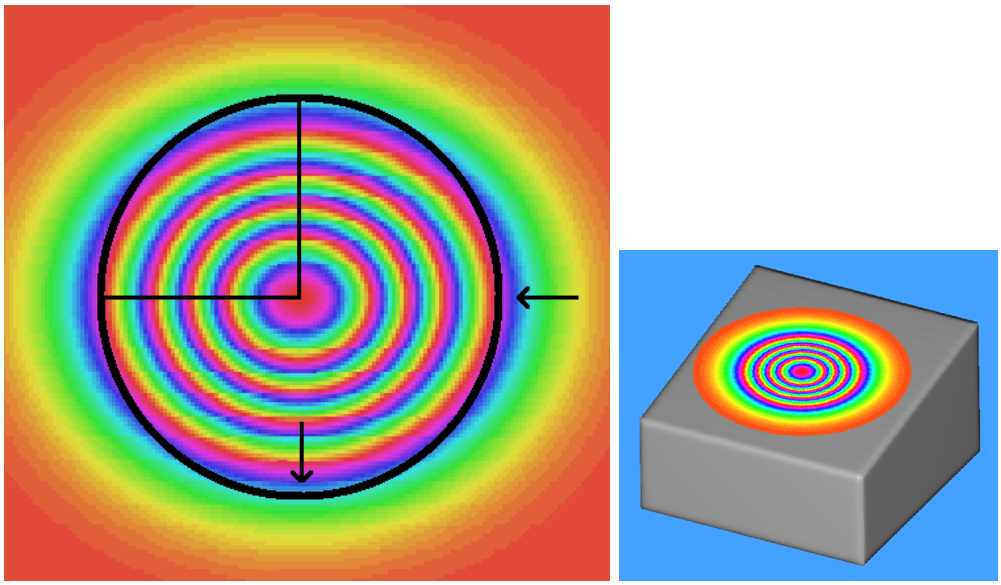


Figure 5: Deviation between the isocurves of an impulse response and the eudlidean isolines on a digital plane with normal $(2, 11, 0)$.

5.1 The first order derivative filters

We introduce here two directional derivative masks which may be used with the convolution operator to obtain two orthogonal differentiation operators.

For each surfel s of Σ we define a numbering of the surfel vertices and edges as illustrated by Figure 2(c), following the coherent orientation around the outward normal. We denote by $E_i(s)$ the e-neighbor of s that shares with s its i^{th} edge. Then, we define the derivative masks $D_{\text{u}}(x, y)$ and $D_{\text{v}}(x, y)$ for $x, y \in \Sigma$ as follows:

$$D_{\text{u}}(x,y) = \begin{cases} \frac{1}{2} & \text{if } y = N_0(x), \\ -\frac{1}{2} & \text{if } y = N_2(x), \\ 0 & \text{otherwise.} \end{cases} \qquad D_{\text{v}}(x,y) = \begin{cases} \frac{1}{2} & \text{if } y = N_1(x), \\ -\frac{1}{2} & \text{if } y = N_3(x), \\ 0 & \text{otherwise.} \end{cases}$$

Note that these two masks correspond to *centered forms* of directional finite differences (i.e., $[-\frac{1}{2}\ 0\ \frac{1}{2}]$). We may also define the masks following the *forward* differences ([0 -1 1]) as follows:

$$D'_{\text{u}}(x,y) = \begin{cases} 1 & \text{if } y = N_0(x), \\ -1 & \text{if } y = x, \\ 0 & \text{otherwise.} \end{cases} \qquad D'_{\text{v}}(x,y) = \begin{cases} 1 & \text{if } y = N_1(x), \\ -1 & \text{if } y = x, \\ 0 & \text{otherwise.} \end{cases}$$

Given the derivative masks D_{u} and D_{v} (resp. D'_{u} and D'_{v}), we may define two derivative operators $\Psi_{f,D_{\text{u}}}$ and $\Psi_{f,D_{\text{v}}}$ ($\Psi_{f,D'_{\text{u}}}$ and $\Psi_{f,D'_{\text{v}}}$) which act on a function f defined on Σ.

Normal vector estimation

Using the iterated convolution operator Ψ, the averaging mask W_{avg}, and the derivative masks D_{u} and D_{v} we define a function $\Gamma^{(n)} : \Sigma \longrightarrow \mathbb{R}^3$ for $n \in \mathbb{Z}$ such that $\Gamma^{(n)}(s)$ is the *estimated normal vector* of Σ at the center of s (after n on-surface convolutions). We define the function $\Gamma^{(n)}$ for $n \in \mathbb{Z}$ by:

$$\Gamma^{(n)}(s) = \frac{\Delta_u^{(n)}(s) \wedge \Delta_v^{(n)}(s)}{\|\Delta_u^{(n)}(s) \wedge \Delta_v^{(n)}(s)\|} \text{ with } \Delta_u^{(n)} = \Psi_{\Psi_{\sigma,W_{\text{avg}}}^{(n)},D_{\text{u}}} \text{ (resp. for } \Delta_v^{(n)}) \tag{1}$$

where $\wedge$ denotes the cross product of two vectors. As the initial averaging process is iterated, the size of the neighborhood taken into account grows accordingly and the precision of the estimate increases as we get closer to the optimal number of iterations. (This number is investigated experimentally in [2, Section 4.3].) We may also define in the same way the functions $\Delta_u'^{(n)}$, $\Delta_v'^{(n)}$ and $\Gamma'^{(n)}$ by using the masks D'_{u} and D'_{v} instead of D_{u} and D_{v}.

The precision of the estimated normals on spheres and tori with descreasing digitization steps has been studied experimentally in [2]. For each digitization step we determined the optimal number of iterations that yielded the smallest average angular error between the estimated and the exact normals. We also conducted experiments in this paper to establish, still experimentally, a link between the maximal curvature of a sphere or a torus and the optimal number of iterations.

As an illustration, an image of a digital object–whose surface has been shaded by a classical method using the estimated normal vectors–is given in Figure 6.

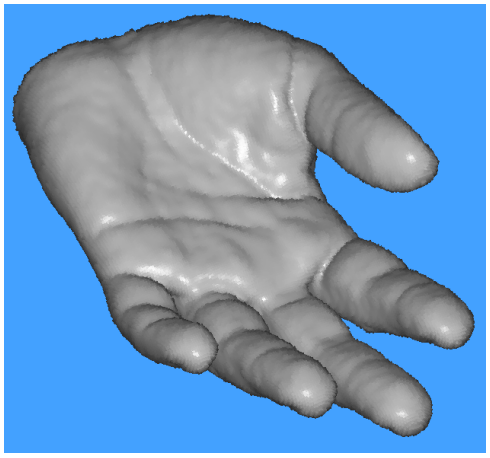


Figure 6: Surface shading of a digitized hand using the normals estimated with our method. The volume size is 149×123×255 and the surface is made of 260,288 surfels. (It was obtained by 3D rasterization of the mesh "Olivier hand" from the AIM@SHAPE repository.)

5.2 Second derivatives operators

Because the masks D_{u} and D_{v} defined in section 5.1 do not generally preserve the orientations of the differentiations when applied to one surfel and one of its neighbors, they should not be applied iteratively to compute second derivatives. Hence, we need to define new differentiation operators.

First, given a surfel s and its i^{th} edge we denote by $EE_i(s)$ the e-neighbor of $E_i(s)$ which is not included in a loop containing s. Intuitively, $EE_i(s)$ is the next surfel one encounters on the surface after $E_i(s)$ when traveling in the direction from s to $E_i(s)$. Furthermore, given the j^{th} vertex of s we denote by $EV_i^j(s)$ the e-neighbor of $E_i(s)$ that shares with s its vertex j. (See Figure 2(c) for the vertices and edges ordering.)

Eventually, for $n \in \mathbb{Z}$ and $s \in \Sigma$ we define

$$\begin{aligned}
\Delta_{uu}^{(n)}(s) &= \frac{\big(\tilde{\sigma}^{(n)}(EE_0(s)) - \tilde{\sigma}^{(n)}(s)\big) - \big(\tilde{\sigma}^{(n)}(s) - \tilde{\sigma}^{(n)}(EE_2(s))\big)}{4} \\
\Delta_{vv}^{(n)}(s) &= \frac{\big(\tilde{\sigma}^{(n)}(EE_1(s)) - \tilde{\sigma}^{(n)}(s)\big) - \big(\tilde{\sigma}^{(n)}(s) - \tilde{\sigma}^{(n)}(EE_3(s))\big)}{4} \\
\Delta_{uv}^{(n)}(s) &= \frac{\big(\tilde{\sigma}^{(n)}(EV_0^0(s)) - \tilde{\sigma}^{(n)}(EV_0^1(s))\big) - \big(\tilde{\sigma}^{(n)}(EV_2^3(s)) - \tilde{\sigma}^{(n)}(EV_2^2(s))\big)}{4}
\end{aligned}$$

where $\tilde{\sigma}^{(n)} = \Psi^{(n)}_{\sigma,W_{\text{avg}}}$. (Remind that $\sigma(s)$ is the center of the surfel s.)

The above definition of the second derivatives correspond to the central differences applied twice. Again, we may use an alternative definition using forward differences in the definitions of Δ'_{uu}, Δ'_{vv} and Δ'_{uv} as follows:

$$\begin{aligned}
\Delta_{uu}^{(n)}(s) &= \tilde{\sigma}^{(n)}(E_0(s)) - 2\tilde{\sigma}^{(n)}(s) - \tilde{\sigma}^{(n)}(E_2(s)) \\
\Delta_{vv}^{(n)}(s) &= \tilde{\sigma}^{(n)}(E_1(s)) - 2\tilde{\sigma}^{(n)}(s) - \tilde{\sigma}^{(n)}(E_3(s)) \\
\Delta_{uv}^{(n)}(s) &= \big(\tilde{\sigma}^{(n)}(EV_0^0(s)) - \tilde{\sigma}^{(n)}(E_0(s))\big) - \big(\tilde{\sigma}^{(n)}(E_3(s)) - \tilde{\sigma}^{(n)}(s)\big)
\end{aligned}$$

5.3 Curvature estimation

Given a *regular parametric surface*

$$\begin{array}{rccl}
S: & D \subset \mathbb{R}^2 & \longrightarrow & \mathbb{R}^3 \\
 & (u,v) & \longmapsto & S(u,v) = (x(u,v), y(u,v), z(u,v))
\end{array}$$

where D is an open and connected subset of $\mathbb{R}^2$, the *Gaussian* (K) and *mean* (H) *curvatures* are defined as follows [6]:

$$K = \frac{e.g - f^2}{E.G - F^2} \qquad H = \frac{e.G - 2.f.F + g.E}{2(E.G - F^2)} \tag{2}$$

with

$$E = \frac{\partial S}{\partial u}\frac{\partial S}{\partial u},\ F = \frac{\partial S}{\partial u}\frac{\partial S}{\partial v},\ G = \frac{\partial S}{\partial v}\frac{\partial S}{\partial v},\ e = N \cdot \frac{\partial^2 S}{\partial u^2},\ f = N \cdot \frac{\partial^2 S}{\partial u \partial v},\ g = N \cdot \frac{\partial^2 S}{\partial v^2}$$

where N is the unit normal field over S.

Note that the maximum and minimum curvatures, respectively denoted κ_1 and κ_2 are given by the following relations:

$$\kappa_1 = H + \sqrt{H^2 - K} \quad \kappa_2 = H - \sqrt{H^2 - K}$$

For a given number n of averaging iterations, we approximate $\frac{\partial S}{\partial u}$ as $\Delta_u^{(n)}$, $\frac{\partial S}{\partial v}$ as $\Delta_v^{(n)}$, $\frac{\partial^2 S}{\partial u^2}$ as $\Delta_{uu}^{(n)}$, $\frac{\partial^2 S}{\partial v^2}$ as $\Delta_{vv}^{(n)}$, $\frac{\partial^2 S}{\partial v^2}$ as $\Delta_{uv}^{(n)}$, and N as $\Gamma^{(n)}$ (see (1)). Eventually, we obtain an estimation of the Gaussian or mean curvature for each surfel of a digital surface using (2).

5.4 Experiments

Our experiments show that the curvatures maybe estimated with the method described above, although quite roughly. Unfortunately, the estimation lacks local regularity. Indeed, the computed curvatures may vary largely from one surfel to its neighbors while they should be very close. We found that applying a second stage of averagings on the computed curvatures provides a more consistent result. Being more consistent, the latter can at least be used to characterize the local shape of a digital surface based on the sign of the Gaussian and mean curvatures. Although not exact, the magnitude of the curvatures may also be useful.

In other words, we divided the estimation process in two steps. First, an estimated Gaussian curvature $\tilde{g_1}(s)$ for each surfel $s \in \Sigma$ was computed after n averaging iterations (i.e., using $\Gamma^{(n)}$ and the operators $\Delta_*^{(n)}$ as described before) and we used $\tilde{g} = \Psi_{g_1, W_{\text{avg}}}^{(n)}$ as the actual estimate.

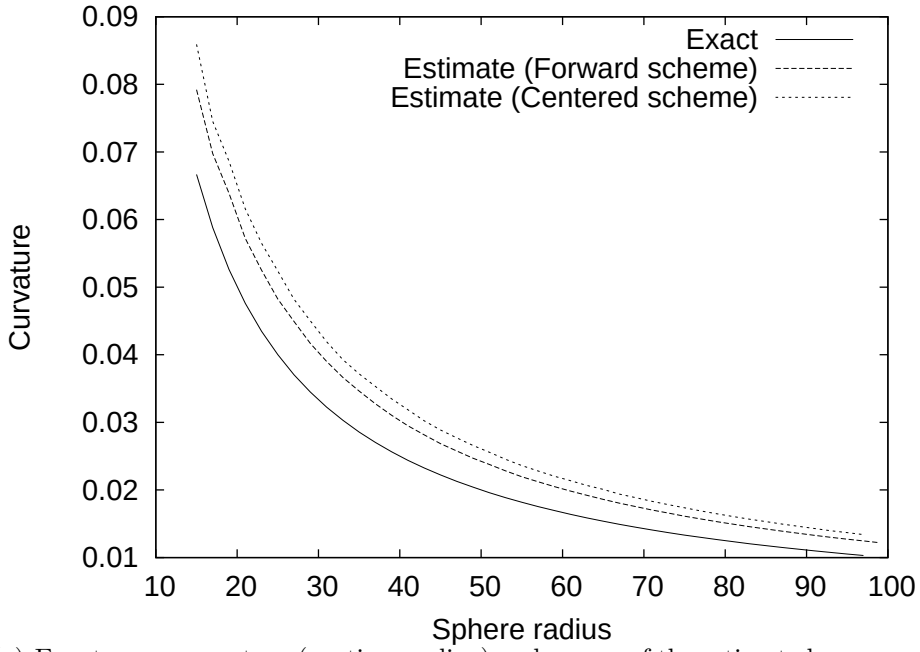
5.4.1 Precision of the estimate on digitized spheres

We have compared the precision of the centered and forward differentiation schemes on digitized spheres. In both cases, we used the optimal number of iterations found experimentally that minimizes the average angular error of the normal estimation (see [2]). We used digitized spheres with radii from 15 to 97 and measured the mean and the standard deviation of the mean curvature over the set of surfels. The result of these measures is depicted on Figure 7.

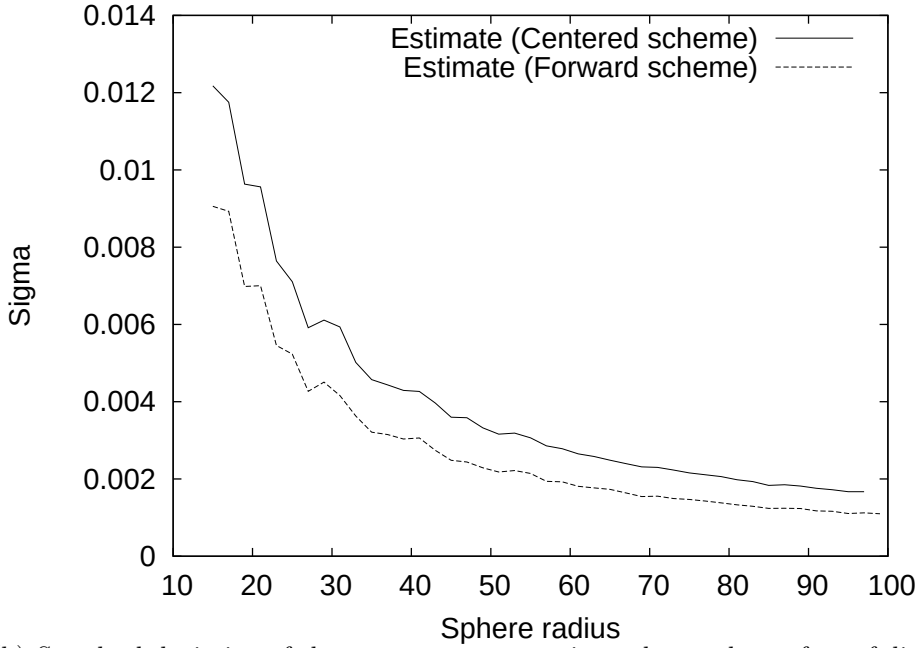
It appears that the curvature is in both cases over-estimated but still close to the actual value. However, this should not be considered satisfactory since the sphere has constant Gaussian and mean curvatures. Although the estimator should at least be tested on this kind of shape, it should of course be tested on surfaces with a not so simple geometry. For that purpose, we carried out some more experiments with tori in the next subsection.

5.4.2 Curvature estimation on a torus

We present here the result obtained on a digitized torus. The torus had a 60 voxels large radius, and a 30 voxels small radius and was rotated along each coordinate axis. Figure 8 shows the estimated and exact Gaussian curvatures for all the surfels sorted according



(a) Exact mean curvature (continuous line) and means of the estimated mean curvatures over digitized spheres with increasing radii. The two dashed lines correspond respectivly to the forward and centered forms of the differentitation schemes.



(b) Standard deviation of the mean curvature estimated over the surface of digitized spheres with increasing radii, using the two differentiation schemes.

Figure 7: Comparision between the centered and forward difference schemes for curvature estimation.

to their increasing exact Gaussian curvatures. The number of iterations n used for the curvature estimation was set according to the results depicted by Fig. 6(a) and 7(a) in [2, Section 4.3], thus it was chosen simply equal to the value of the small radius of the torus. The same number was used in the second stage of averagings mentioned before.

Eventually, the possible use of the estimated curvature for the local characterization of shapes is illustrated by Figure 9. As shown in Figure 9(a), areas of negative and positive Gaussian curvatures are "correctly" localized on a torus. On the other hand, we show in Figure 9(b) that the estimated mean curvature allows to determine the valleys, hills and areas with high curvature on the surface of the Standord bunny.

6 Conclusion

We presented in this report some preliminary results about curvature estimation base on the generalized convolution operator introduced in [2]. Right now, the curvature estimator cannot be qualified as very precise but still can be used for rough shape characterization, like for example the segmentation between regions of negative, positive, high or low curvature ; as suggested by Figure 9.

However, a way to determine the optimal number of convolution iterations that yields the better estimate, for the surface of a given digital object, is still to be found. Furthermore, the fact that the estimate gets more precise when the digitization step decreases has only been tested experimentally on a few types of objects. The two previous remarks were actually made and still hold for the normal estimation presented in [2], and we will also conclude by saying that a formal proof of what is called *multigrid convergence* in the literature should be a difficult task to be addressed in the future.

References

[1] S. Fourey, R. Malgouyres, Normals and curvature estimation for digital surfaces based on convolutions, in: DGCI, vol. 4992 of Lecture Notes in Computer Science, Springer, 2008.

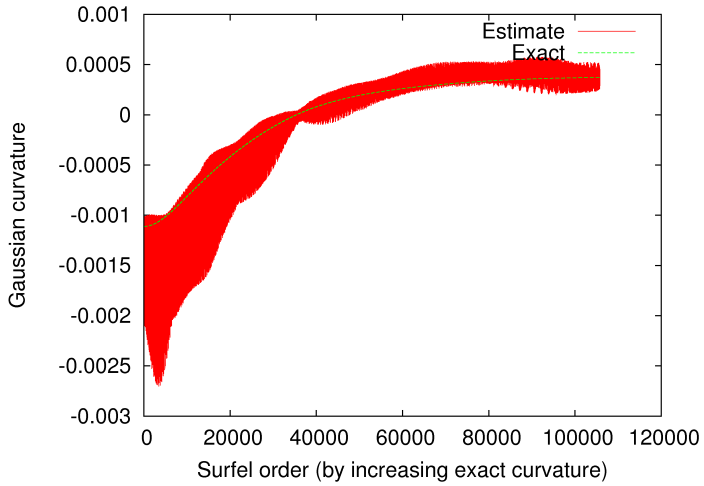
[2] S. Fourey, R. Malgouyres, Normals estimation for digital surfaces based on convolutions, Computers & GraphicsTo appear.

[3] G. T. Herman, Discrete multidimensional Jordan surfaces, CVGIP: Graphical Models and Image Processing 54 (6) (1992) 507–515.

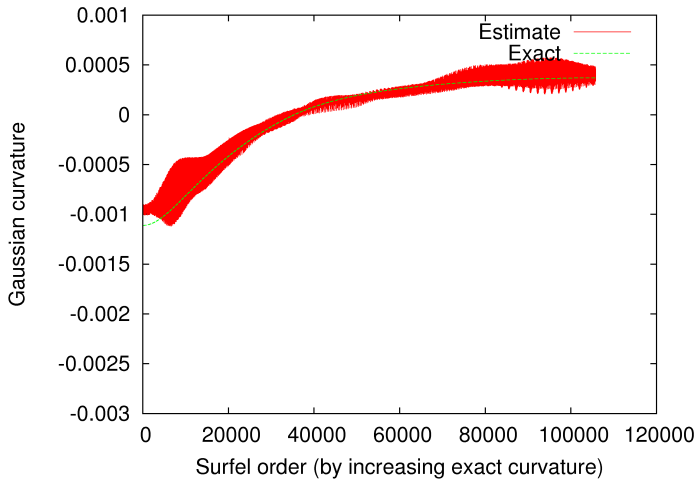
[4] R. Malgouyres, F. Brunet, S. Fourey, Binomial convolutions and derivatives estimation from noisy discretizations, in: Discrete Geometry for Computer Imagery, vol. 4992 of Lecture Notes in Computer Science, 2008.

[5] R. Malgouyres, A. Lenoir, Topology preservation within digital surfaces, Graphical Models 62 (2) (2000) 71–84.

[6] J. J. Stoker, Differential geometry, Wiley, 1989.

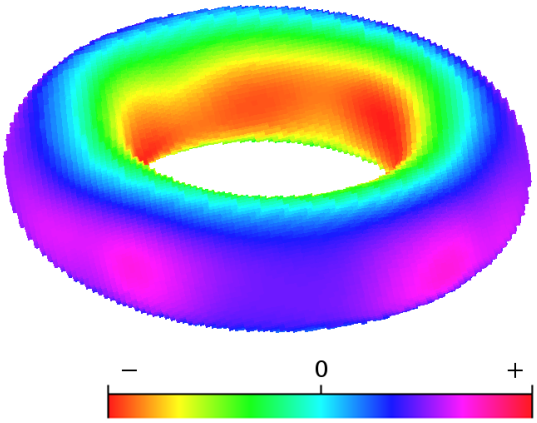


(a) Estimated Gaussian curvatures using the forward differenciation scheme.

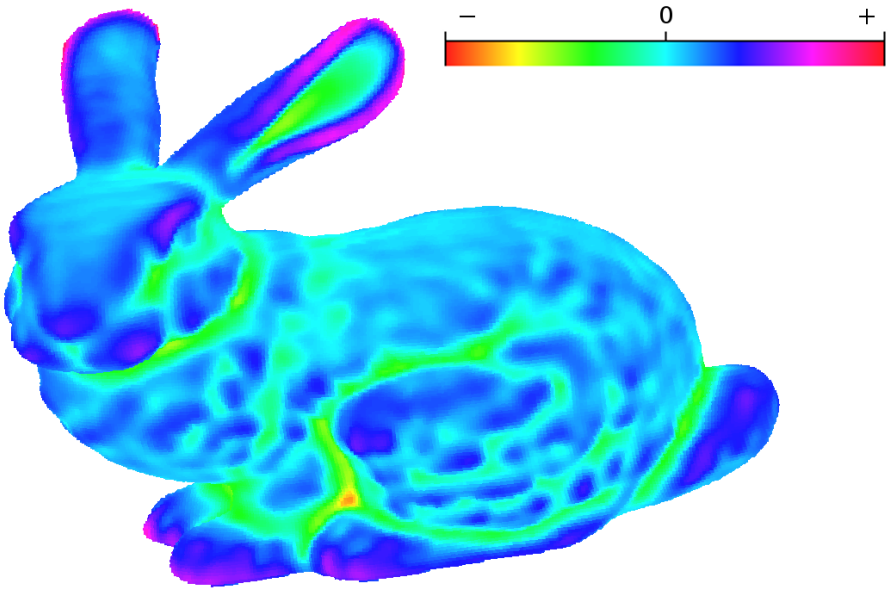


(b) Estimated Gaussian curvatures using the centered differenciation scheme.

Figure 8: Estimated and exact Gaussian curvatures on a torus (large radius is 60, small radius is 30, rotations along the three axes are respectively 114°, 57°, and 22°). Surfels are numbered on the horizontal axis according to their increasing exact Gaussian curvatures. The estimated values appear as a thick area because they are plotted with lines and vary very quickly.



(a) Surface of a torus shaded according to the averaged estimated Gaussian curvature.



(b) The Stanford bunny shaded according to the averaged estimated mean curvature.

Figure 9: Estimated curvature on two objects.