Curvature Estimation for Discrete Curves Based on Auto-adaptive Masks of Convolution

Christophe Fiorio[1], Christian Mercat[12], and Frédéric Rieux[12]

[1] LIRMM, Université Montpellier 2, 161 rue Ada,
F-34392 MONTPELLIER, France
[2] I3M, Université de Montpellier 2 c.c. 51
F-34095 Montpellier Cedex 5, France

Abstract. We propose a method that we call auto-adaptive convolution which extends the classical notion of convolution in pictures analysis to function analysis on a discrete set. We define an averaging kernel which takes into account the local geometry of a discrete shape and adapts itself to the curvature. Its defining property is to be local and to follow a normal law on discrete lines of any slope. We used it together with classical differentiation masks to estimate first and second derivatives and give a curvature estimator of discrete functions.

1 Introduction

The discretization of a smooth object loses some features of the original geometrical characteristics. An important problem is to estimate derivatives of digital functions, for example the tangent space at a point or the curvature of the shape. There exists many approaches based on line segmentation [11, 7, 14] or on filtering [19, 13, 8]. New approaches to derivative estimation [12, 8] are based on convolution product with a diffusion kernel and give striking estimations of derivatives on noisy curves. Those convolutions use binomial kernels which are not adaptive to the local geometry of the discrete object. In [8], a surface kernel has been proposed where the computed weights take into account the local disposition of the adjacent voxels. Ours work builds on this approach.

The main issue is to compute convolution masks best suited for a particular discrete object. We propose kernels that adapt themselves to the geometry. Those masks enable us to give a good estimation of derivatives and curvature of discretized functions.

This article is laid out as follows: We introduce convolution product in Sec. 3 and define our auto-adaptive process in Sec. 4 as a random process defining our convolution kernel. We investigate discrete lines in Sec. 5 and prove that their masks are equivalent to normal laws. We normalize their deviations by allowing some optimized tunnel effect in Sec. 5.3, recovering Euclidean properties for lines of any slopes. Finally we show the usefulness of our approach by giving estimations of derivative and curvature of discretized functions in Sec. 6.

2 Diffusion Processes

Heat kernel or random walks have been widely used in image processing, for example lately by SUN, OVSJANIKOV and GUIBAS [18] and GEBAL, BÆRENTZEN, AANÆS and LARSEN [9] in shape analysis. It is indeed a very precious tool because two manifolds are isometric if and only if their heat kernels are the same (in the non degenerate case).

The heat kernel k_t of a manifold M maps a couple of points $(x, y) \in M \times M$ to a positive real number $k_t(x, y)$ which describes the transfer of heat from y to x in time t. Starting from a (real) temperature T on M, the temperature after a time t at a point x is given by a convolution of the initial temperature distribution with the diffusion kernel:

$$H^t f(x) = \int_M f(y)\, k_t(x, y)\, dy.$$

The heat equation is driven by the diffusion process, the evolution of the temperature in time is governed by the (spatial) LAPLACE-BELTRAMI operator Δ_M: $\frac{\partial f(t,x)}{\partial t} = -\Delta_M f(t, x)$ which presides to the diffusion on the manifold, for example random walks.

The first issue in order to use these ideas in the discrete setup is to define a good discrete Laplacian, or equivalently, a good diffusion process. It should be reasonably robust to noise, to outliers (points which are added by mistake) and to missing data. This situation is understood in the realm of polyhedral surfaces and triangulations, and a time appraised discrete Laplacian, based on sound theoretical grounds is known for a long time, the so-called *cotangent weights* Laplacian [15]. In this article, we define a diffusion process in 2D-Digital Geometry, where the space is composed of pixels.

3   Convolution Products

Estimation of geometrical properties is an important goal of discrete geometry. There exists many methods to estimate the length [4], or the curvature of discrete curves [11, 7, 14, 3] such as the discretized graph of a function. A method is to recognize maximal straight segments [6] but it is sensitive to noise. Our aim is to extend the pioneer work of [8] on product convolution technique to produce *adaptive averaging masks* for an arbitrary connected set of pixels representing the quantization of a curve in the plane. Commonly, the convolution product *smooths* a function g by an averaging function f, usually concentrated around the origin which slides along the line, $\mathbb{Z}$ in the discrete setup: $f * g(x) = \sum_{t \in \mathbb{Z}} f(x - t).g(t)$.

In the case of functions defined on a less homogeneous set of pixels E, no simple addition is defined, so we replace the sliding function by a 2-points kernel of limited support (a *mask*):

Definition 1 (Mask product).
The product $f * g$ of a kernel $f \; : \; E \times E \to \mathbb{R}$ and a discrete function $g \; : \; E \to \mathbf{F}$ to a vector space $\mathbf{F}$, is given by:

$$f * g : \; E \to \mathbf{F} \tag{1}$$
$$x \mapsto \sum_{y \in \mathbf{E}} f(x, y).g(y). \tag{2}$$

Convolution with itself widens the mask's range: $f^{(n)} * g = f * (f^{(n-1)} * g)$. The averaging kernel $\boxed{\;|\tfrac{1}{2}|\tfrac{1}{2}|\;}$ is for example widened into the binomial $\frac{1}{2^n} \boxed{\;|1|n|} \cdots \boxed{\binom{n}{p}} \cdots \boxed{|n|1|\;}$.

These definitions can be extended to the 2D case where such convolution masks are used in noisy digital objects, blurring sharp details but otherwise limiting the effect of noise.

In this work we introduce masks whose weights are suitable for any 1D-curve and any shape. Rémy MALGOUYRES and Sébastien FOUREY [8] defined mask weights for discrete surfaces in 3D-space that are computed taking into account the neighbors of the voxels. Likewise, our masks are

computed according to the local geometry of a set of pixels. Those masks are adaptive and allow us to smooth a function with suitable weights for the studied curve. They are based on auto-adaptive diffusion processes.

4 Auto-adaptive Process

We setup the diffusion of a walker wandering on the discrete object, in order to recover its local geometry. This auto-adaptive process is a local MARKOV process, a stochastic process in discrete time where the probability of transition between pixels depends only on the immediate past. The walker goes from a pixel to the next through the corners they have in common.
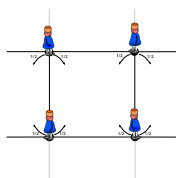
Definition 2 (Standard adaptive process).
We call standard adaptive process*, the* MARKOV *chain with the probabilities of transition from pixel to pixel defined by the composition of transitions from a pixel of the set to its four corners and back from a corner to adjacent pixels of the set constrained by:*
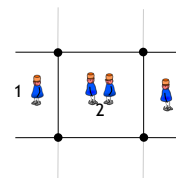
1. *Equiprobability $\frac{1}{4}$ to go from a pixel to one of its corners;*
2. *Equiprobability to move from a corner to an adjacent pixel of the set.*
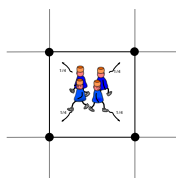
| (a) From pixel to corners. | (b) From corners to adjacent pixels. | (c) On $\mathbf{Z}$, the binomial coefficients are recovered. |

Fig. 1: A walker has a probability of $\frac{1}{4}$ to pick up a corner of a pixel, then $\frac{1}{k}$ to select a pixel among k possible, ending up in the binomial coefficients $\{1, 2, 1\}$ for the random walk on $\mathbf{Z}$.

This standard choice amounts to walkers with no memory and only local knowledge, the celebrated short-sighted drunken man. We note this process A_s^m for a walker starting at any given point, with m the number of iterations of the process.

Remark 3. A_s^m is the stochastic matrix of the studied discrete object. We obtain a mask of length $2m+1$ centered on the pixel i if we multiply the matrix A_s^m by a position vector $v_i = (0, \ldots, 0, \underbrace{1}_{i-th\ position}, 0, \ldots, 0)^T$.

In the sequel, we call v_x the position vector of the pixel x.

We illustrate the definition with examples (see Figure 2b) of masks computed on discrete lines. After m steps the process builds a mask of length $2m + 1$.

We call our process *standard* because of the equiprobability condition. In the next section, we will modify these probabilities to uniformize the diffusion on discrete lines. Note that the standard adaptive process on a finite connected discrete object is *ergodic*.

On the integer line $\mathbb{Z}$, the diffusion leads to the Gaussian binomial masks $A_s^n(x, y) = \binom{n}{|x-y|}$. We prove next that on discrete lines, it is a more involved Gaussian mask.

5 Convolution Masks on Discrete Lines

We investigate here the auto-adaptive standard masks on discrete lines of different slopes.

We recall the definition of arithmetics discrete lines and we prove that the weights of the associated auto-adaptive masks have a Gaussian distribution. But their deviations differ for different slopes.
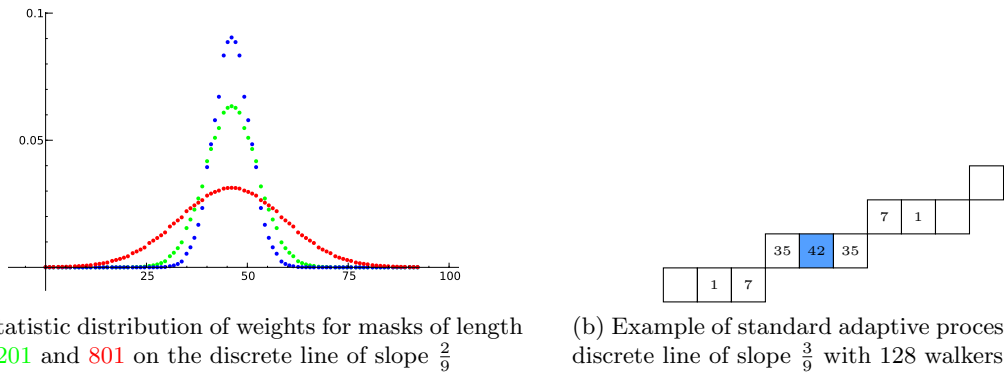
5.1 Auto-adaptive Masks on Discrete Lines

Definition 4 (Discrete line [16]).
A discrete line of slope $\frac{a}{b}$ with $a \wedge b = 1$, inferior limit μ and arithmetic thickness ω is the set of pairs $(x, y) \in \mathbb{Z}^2$ such that

$$\mu \leq ax - by < \mu + \omega$$

with a, b, ω and μ integers.

In discrete geometry there are two adjacencies commonly used, the connectivity by edge, called 4-adjacency and the connectivity by vertex, called 8-adjacency. We are interested in naive lines ($\omega = \max(|a|, |b|)$), i.e. the thinner 8-adjacent connected lines for a given slope.

(a) Statistic distribution of weights for masks of length 101, 201 and 801 on the discrete line of slope $\frac{2}{9}$

(b) Example of standard adaptive process on the discrete line of slope $\frac{3}{9}$ with 128 walkers.

Fig. 2: Examples of walkers diffusion on discrete lines.

Example results in Figure 2a show that for a mask large enough, the statistic distribution can be estimated by a continuous normal law with the same mean and deviation. The proof of this observation resides on ergodicity and the application of the Central Limit Theorem. We first need a technical definition:

Definition 5 (Equivalence class).
We assign to a pixel i the pair of adjacencies of its left and right neighbors and denote $[i]$ its induced equivalence class.

Theorem 6. *The standard auto-adaptive process A_s^m computed on a discrete line $\mathcal{D}(a, b, \mu, \omega)$ is equivalent to a normal law when $m \to \infty$.*

See its proof in the appendix on p.12.

4

$$\boxed{\tfrac{1}{4}}\ \boxed{\tfrac{1}{2}}\ \boxed{\tfrac{1}{4}}$$

(a) 4-4

(b) 8-8

(c) 8-4

(d) 4-8

Fig. 3: There are only four equivalence classes appearing on an 8-connected line in the first octant.

5.2 Curvilinear Abscissa Versus Integer Index

The standard adaptive process gives us good normal laws but they are not completely satisfactory because their deviation depends on the line slope: pixels which are only 8-connected act as filters (see Fig. 4b). We measure the dispersion of walkers on a discrete line to compare the dispersion between two discrete lines of different slopes. In Fig. 4b, we have used a discrete index i with respect to which we computed the spatial deviation. It is a relevant index for *graphs* of discrete functions $f : \mathbb{Z} \to \mathbb{Z}$, but it is not for arbitrary curves. We need to take into consideration the curvilinear abscissa of the underlying curve. This abscissa is easy to compute for lines with the help of projection (see Fig. 9a) on p. 7.

The difference between the index and the curvilinear abscissa can be as big as a speeding factor $\sqrt{2}$ for diagonal lines, which are no longer the minimum of this curvilinear deviation (see Fig. 4b).
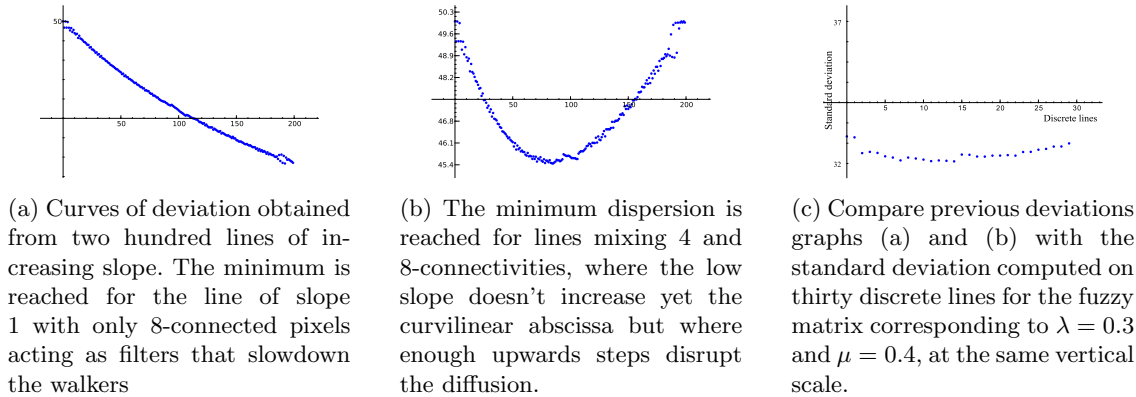
(a) Curves of deviation obtained from two hundred lines of increasing slope. The minimum is reached for the line of slope 1 with only 8-connected pixels acting as filters that slowdown the walkers

(b) The minimum dispersion is reached for lines mixing 4 and 8-connectivities, where the low slope doesn't increase yet the curvilinear abscissa but where enough upwards steps disrupt the diffusion.

(c) Compare previous deviations graphs (a) and (b) with the standard deviation computed on thirty discrete lines for the fuzzy matrix corresponding to $\lambda = 0.3$ and $\mu = 0.4$, at the same vertical scale.

Fig. 4: Standard deviations of masks for discrete lines of different slopes with different methods.

5.3 Standardization of Standard Deviation for any Slope

To balance the effect of digitization which creates narrow links for diagonal lines and links for horizontal and vertical lines, we will allow the walker to step *outside* the discrete line. We thicken discrete lines and we modify the probability transition in this thick virtual, fuzzy part. We consider three natural sets associated with a thin discrete line, the line itself (see Fig. 5), a thicker one obtained by adding all the 4-adjacent pixels (see Fig. 6), and a larger one by adding all the 8-adjacent pixels (see Fig. 7).

5

By definition of discrete lines, we could increase the arithmetic thickness ω. But ω depends on the discrete line under study, and the number of pixels added depends on the slope as well. We want, on the contrary, to define a local thickening procedure without taking into account global features like the arithmetic thickness, to apply it on arbitrary set.

We build this way three MARKOV processes, associated with matrices, M_0, M_1 and M_2 of increasing dimensions by adding all the pixels neighbors.
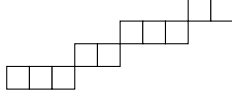

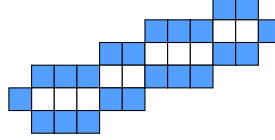Fig. 5: Discrete standard line of slope $\frac{2}{5}$

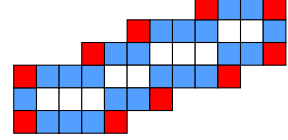Fig. 6: Add the 4-connected pixels of the line

Fig. 7: Add the 8-connected pixels of the line

Definition 7 (Fuzzy Matrix).
Let M_0, M_1 and M_2 the matrices of the discrete lines respectively the naive thin line, its thickened version with the addition of 4-connected neighbors and the fatter line with 8-connected neighbors. Let M_0' and M_1' the matrices induced by M_0 and M_1, padded by columns and rows of zeros and a diagonal of 1 for pixels not belonging to these sets, so that the three matrices have the same dimension as M_2. We define F a function with two parameters λ and μ defined by:

$$F \ : \ (\lambda, \ \mu) \ \mapsto \ (1 \ - \ (\lambda \ + \ \mu))M_0' \ + \ \lambda M_1' \ + \ \mu M_2'$$

We call F the fuzzy matrix of parameters λ and μ.

(a) 4-8 thickened

(b) 8-4 class thickened
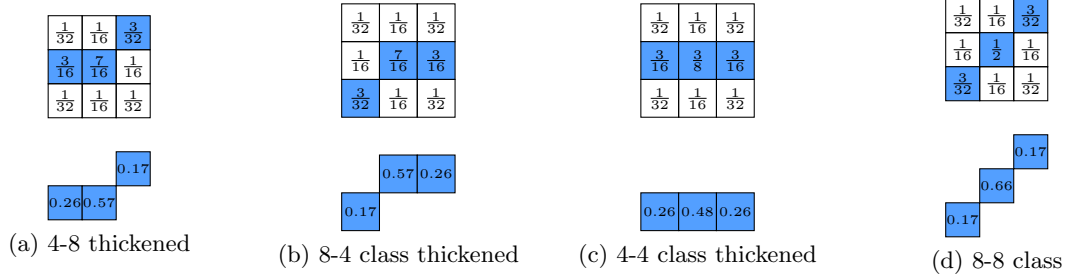
(c) 4-4 class thickened

(d) 8-8 class

Fig. 8: Computed projection masks of the four equivalent classes on naive discrete lines (Fig. 3). First, we compute the thickened mask of the fuzzy matrix for $\lambda = 0$, $\mu = 0.5$, close to the minimum dispersion. In white we have the ghost pixels, in blue the primary points. Then, to give an example we project on the line $2x - 5y = 0$ to obtain the final mask on the line.

The fuzzy matrix is still associated with a stochastic process but on a larger set of pixels: we allow the walker to go outside the main discrete line so the process, when restricted to the discrete

line, is no longer stochastic, because at the end of the process, many walkers appear in *ghost* pixels. We bring them back on line by a simple projection onto the associated real line of the same slope, as illustrated in Figure 9a. We obtain two sets of points; the first corresponds to points belonging to the original naive discrete line called *primary points*, and the others correspond to the ghost points (see Fig. 8). We affect ghost walkers to primary points according to the distance between their projection point and the two nearest primary points (see Figure 9b). We do not lose any walker and we take into account the position of their ghost pixels.
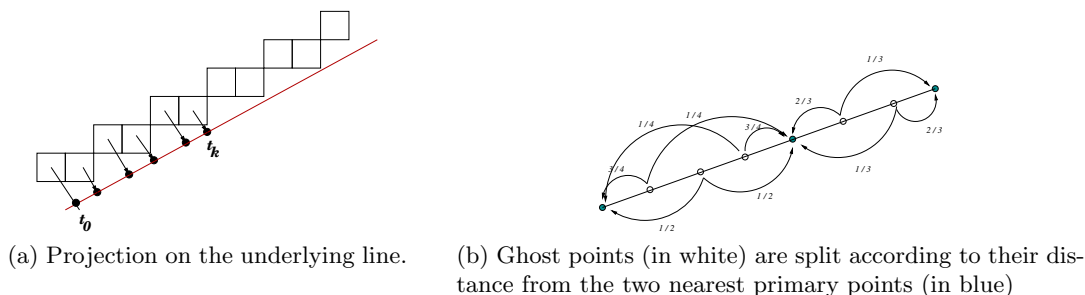

(a) Projection on the underlying line.

(b) Ghost points (in white) are split according to their distance from the two nearest primary points (in blue)

Fig. 9: Projection of walkers on the underlying line and repartition onto primary points.

For fixed λ and μ, we computed the deviations of our fuzzy diffusion process on lines of increasing slopes and computed the standard deviation of these deviations. We plotted this dispersion as a function of λ and μ (see Fig. 10a). We found experimentally the best values for λ and μ to have minimal dispersion.

(a) Computation of the values of the dispersion of standard deviations as a function of λ and μ.

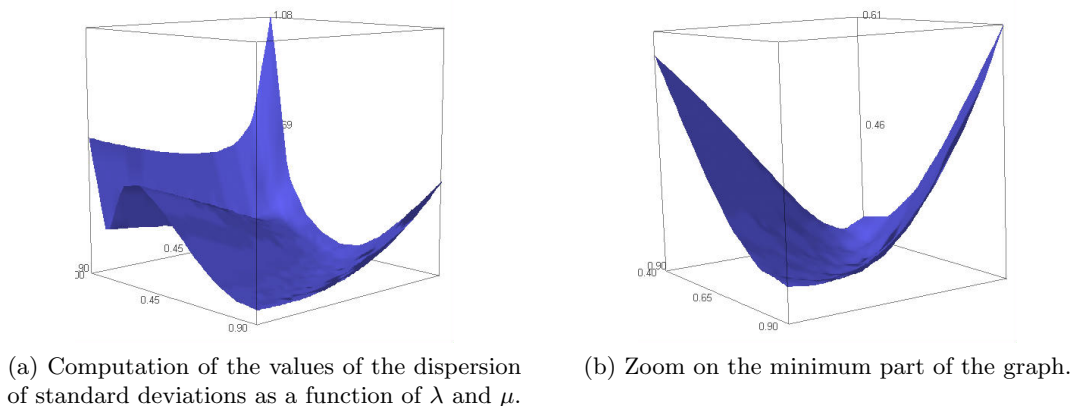(b) Zoom on the minimum part of the graph.

Fig. 10: We get a minimal dispersion for the values $\lambda = 0.3$ and $\mu = 0.4$.

The general shape of this dispersion and the location of its minima don't depend much on the size of the mask once it is big enough (a few pixel wide) and on the number of lines (we used up to

50 lines): We found (see Figure 10a) $\lambda = 0.3$ and $\mu = 0.4$, with a long valley of acceptable values up to $\lambda = 0$, $\mu = 0.5$. In particular, good values are given when there are not much differences in probability between the 4 and 8-connected added neighbors.

As we can see in Figure 4c, the fuzzy process with suitable λ and μ gives masks with approximately the same standard deviation for any line.

With these values, we have now a local fuzzy process, which is adaptive to the shape of the underlying set and which has been tailored to recover the Euclidean geometry of discrete lines.

Therefore, given a curve, which is by definition locally modeled on a line, its discretization with a step finer than its feature size, can be analyzed by our mask, of a proper standard deviation between the discretization step and the feature size, and give back the Euclidean geometry of the curve. We are going to show in the next section that it indeed allows us to compute derivative and second derivative of discrete functions with great accuracy.

6 Derivative and Curvature Estimation

Fuzzy process are expensive in terms of computation because we need to increase significantly the number of points of the studied object. We will show that, in order to compute derivatives of discretized functions, the standard auto-adaptive kernel, which is cheaper, is already very efficient. But fuzzy processes are clearly the most suitable and we can improve the results. We will study tangent estimation and we will show our results on the function $x \mapsto \sin(x)$.

First we will introduce how we obtained the set of studied pixels for the discretized graph of a real function, then we will show how we compute the derivation kernel according to the standard auto-adaptive process on a digital curve. Finally we will give a curvature estimator and we will display the results computed on the discrete sinus function.

6.1 Derivative Function Estimation

We use a classical method of discretization called grid intersect quantization (GIQ) [17].

In order for this procedure to give us a connected set of pixels, we rely here on a pre-processing that *flattens* the function in order to have a derivative of magnitude less than 1. We will describe in a future article how to deal with supercover graphs [1].

Let φ be a digital function defined on $I \subset \mathbb{Z}$, and $A_s^m(\varphi)$ the fuzzy process on its digital graph. We want to approximate the derivative function φ' according to the discrete values of φ.

There are three popular discrete derivation masks: the centered δ_0 : $\begin{array}{|c|c|c|c|c|}\hline & \frac{1}{2} & & -\frac{1}{2} & \\\hline\end{array}$, the right-sided δ_+ : $\begin{array}{|c|c|c|c|c|}\hline & & 1 & -1 & \\\hline\end{array}$ and the left-sided δ_- : $\begin{array}{|c|c|c|c|c|}\hline & 1 & -1 & & \\\hline\end{array}$. We will specify for each estimation the mask used. Our convolution kernel $A_s^m(\varphi)$ smooths the function φ, before and after the application of the derivation operator D_1.

Definition 8 (Derivative estimator). *Let φ be a discrete function and $A_s^m(\varphi)$ be the kernel of the auto-adaptive process. Then the estimated derivative of the function φ at the point x is given by:*

$$D_1(x) = [A_s^m(\varphi) * \delta_* * A_s^m(\varphi)](x)$$

with δ_* one of the three derivation kernels.

The convolution product is not commutative, and we can increase several times the support of the mask to the right or the left of D_1 in order to raise the smoothing effect of the kernel and improve the estimation on noisy functions to the price of an erosion of the small scale data. As an example in Fig. 11, we give an estimation of the derivative function $x \mapsto \sin(x)$ with the centered kernel δ_0.
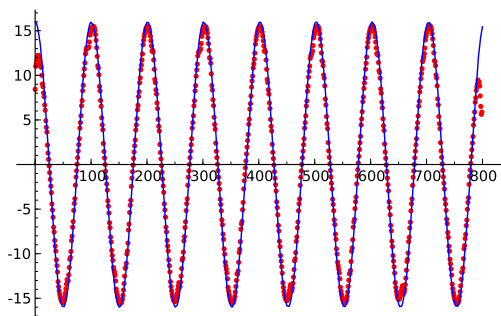
(a) Estimation of the discrete adaptive derivative function of $x \mapsto sin(x)$ and the values of the real function $x \mapsto cos(x)$, computed according a mask of length 61. We haven't hidden the boundary effects. The erosion is mild.
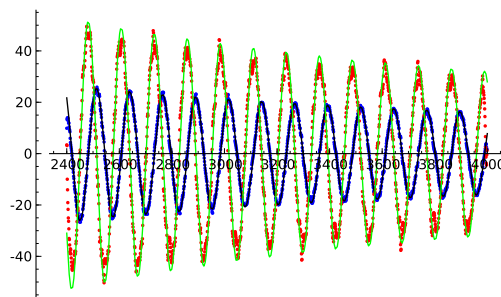
(b) Estimation of the discrete derivative function of $x \mapsto \frac{20}{x} sin(2x)$ of order one and two, computed according a mask of length 30.

Fig. 11: Estimation of derivative for different functions.

6.2 Curvature Estimation

We define a derivation kernel to compute derivatives of higher order used in the curvature.

Definition 9 (Derivative of higher order). *Let φ be a discrete function and $A_s^m(\varphi)$ be the auto-adaptive process computed on φ. Let δ_* be a derivation kernel, then the derivation kernel of order n is defined as*

$$D_n(x) = \left[A_s^m(\varphi) \underbrace{* \delta_* * A_s^m(\varphi) * \ldots * A_s^m(\varphi)}_{n \ times} \right] (x)$$

As an example we give the estimated second derivative of $x \mapsto \sin(x)$ (see Fig. 12a) for the centered derivation kernel.

We recall the classical definition of the curvature for a real function in the plane. Let $(x,\ y(x))$ be a function graph with $x \in I \subset \mathbb{R}$. Then the curvature is given by $\Gamma(x) = \frac{y''(x)}{(1\ +\ y'^2(x))^{\frac{3}{2}}}$. The curvature estimation for a discrete function is based on this formula and the estimation of φ' and φ'':

$$\Gamma(x) = \frac{D_2(x)}{(1\ +\ D_1(x)^2)^{\frac{3}{2}}}.$$

9

(a) Comparison between the values of $x \mapsto \sin(x)$, its discretization (in green), its continuous first and second derivative (in blue and black), and their estimations with the kernel A_s^{10}.

(b) Comparison between the real (blue) and discretised (red) curvatures of the function shown in Fig. 12a.
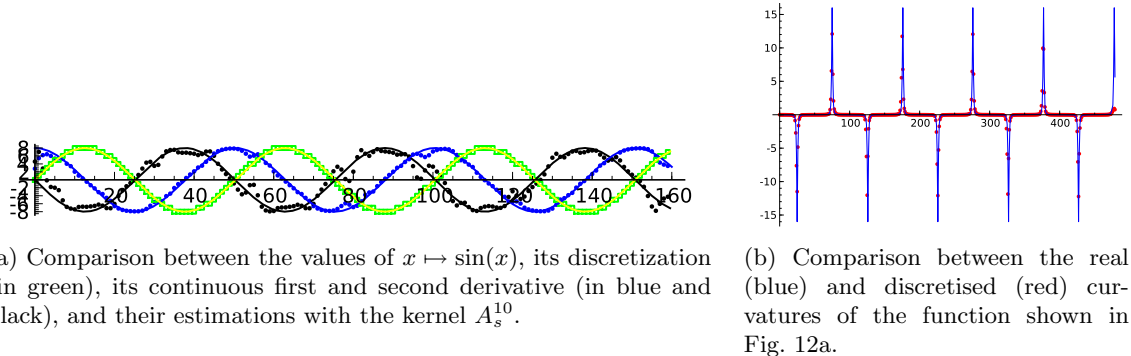
Fig. 12: Second order derivative and curvature are amazingly accurate (mask of size 10). A small mask gives good estimations although artifacts are visible, shifts on the first derivative (around 30, 50, 80 and 100 for example) and erosion on the second order derivative.

We computed the estimated curvature of the discretised function $x \mapsto \sin(x)$ and we compared it with the real curvature. As we can see in Fig. 12b, for a typical mask of convolution, we obtain a curvature close to the exact curvature.

The result is encouraging because we recover, in spite of the noise induced by the discretization, a good estimate of the curvature. Moreover, experiments reveal that our estimator gives results comparable to many other estimators: Tangent estimators based on recognition discrete lines [11, 7, 14] are sensitive to noise and our Gaussian approach should not. On the other hand, methods based on filtering [19, 13] use convolution masks which are not adaptive to the local geometry. Our approach is more faithful to the discrete object and is versatile: we can intensify the smoothing or reduce it according to prior knowledge on the data or revise the data once the local feature size has been roughly estimated. The performance of the fitting in comparison to the size of the masks, the local feature size and the intensity of the final convolution will be studied in a forthcoming paper.

7   Conclusion

We defined an averaging operator based on diffusion on a discrete set. The distribution of weights on discrete lines is equivalent to a normal law and we standardize its diffusion to have approximately the same standard deviation for any slopes, recovering their Euclidian geometry. The computed masks smooth out correctly discretisation noise, and provide a first step in recovering the intrinsic 1D-geometry of the underlying object: convolved with differentiation kernel, we build a derivative estimator that we use to estimate tangent vector and curvature.

This estimator will be investigated in the future to account for discretised functions with non-connected discretised function graphs. More generally, we will describe in a forthcoming paper how to define a parameterisation of discrete curves by curvilinear abscissa for an arbitrary discrete set modeled on an unknown 1D curve. We will then generalize our method to digital surfaces in 3D.

References

1. E. Andres. *Modélisation Analytique Discrète d'Objets Géométriques.* Habilitation à diriger des recherches, UFR Sciences Fondamentale et Appliquées - Université de Poitiers (France), Décembre 2000.

2. Patrick Billingsley. *Convergence of probability measures.* Wiley Series in Probability and Statistics: Probability and Statistics. John Wiley & Sons Inc., New York, second edition, 1999. A Wiley-Interscience Publication.

3. David Coeurjolly, Isabelle Debled-Rennesson, and Olivier Teytaud. Segmentation and length estimation of 3d discrete curves. In *Digital and Image Geometry*, pages 299–317, 2000.

4. David Coeurjolly and Reinhard Klette. A comparative evaluation of length estimators of digital curves. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(2):252–257, 2004.

5. David Coeurjolly, Isabelle Sivignon, Laure Tougne, and Florent Dupont, editors. *Discrete Geometry for Computer Imagery, 14th IAPR International Conference, DGCI 2008, Lyon, France, April 16-18, 2008. Proceedings*, volume 4992 of *Lecture Notes in Computer Science.* Springer, 2008.

6. Isabelle Debled-Rennesson and Jean-Pierre Reveillès. A linear algorithm for segmentation of digital curves. *IJPRAI*, 9(4):635–662, 1995.

7. Fabien Feschet and Laure Tougne. Optimal time computation of the tangent of a discrete curve: Application to the curvature. In *DGCI*, pages 31–40, 1999.

8. Sébastien Fourey and Rémy Malgouyres. Normals and curvature estimation for digital surfaces based on convolutions. In Coeurjolly et al. [5], pages 287–298.

9. Katarzyna Gebal, J. Andreas Bærentzen, Henrik Aanæs, and Rasmus Larsen. Shape Analysis Using the Auto Diffusion Function. In Konrad et al. [10], pages 1405–1413.

10. Polthier Konrad, Alexa Marc, and Kazhdan Michael, editors. *Symposium on Graphics Processing.* Eurographics Association, 2009.

11. Jacques-Olivier Lachaud, Anne Vialard, and François de Vieilleville. Fast, accurate and convergent tangent estimation on digital contours. *Image Vision Comput.*, 25(10):1572–1587, 2007.

12. Rémy Malgouyres, Florent Brunet, and Sébastien Fourey. Binomial convolutions and derivatives estimation from noisy discretizations. In Coeurjolly et al. [5], pages 370–379.

13. Jiri Matas, Z. Shao, and Josef Kittler. Estimation of curvature and tangent direction by median filtered differencing. In *ICIAP*, pages 83–88, 1995.

14. Thanh Phuong Nguyen and Isabelle Debled-Rennesson. Curvature estimation in noisy curves. In *CAIP*, pages 474–481, 2007.

15. Ulrich Pinkall and Konrad Polthier. Computing discrete minimal surfaces and their conjugates. *Experiment. Math.*, 2(1):15–36, 1993.

16. J.-P. Reveillès. *Géométrie discrète, Calcul en nombres entiers et algorithmique.* PhD thesis, Université Louis Pasteur, Strasbourg, France, 1991.

17. A. Rosenfeld. Digital straight line segments. *IEEE Transactions on Computers*, 23(12):1264–1269, 1974.

18. Jian Sun, Maks Ovsjanikov, and Leonidas Guibas. A Concise and Provably Informative Multi-Scale Signature Based on Heat Diffusion. In Konrad et al. [10], pages 1383–1392.

19. M. Worring and A. W. Smeulders. Digital curvature estimation. *CVGIP: IMAGE UNDERSTANDING*, 58(3):366–382, 1993.

Appendix Proof of Theorem 6

Proof. Contrary to the case on $\mathbb{Z}$ the random variable $X_t \in \{-1, 0, +1\}$ that drives the wandering at time t, (left, stay put or right), is not equal for each pixel, hence for each time. These pixels are distributed in few classes determined by the ring of pixels nearby (see Definition. 5 on p. 4). The position evolution at time t, $P_t = P_{t-1} + X_{P_{t-1}}$ brings into play the random variable X_t that depends on t through the position P_{t-1}. We can write it in the form of:

$$\frac{1}{T} P_T = \frac{1}{T} \sum_{t=1}^{T} X_t$$

Since the system is *ergodic* [2], this time average of X along a trajectory up to T is equal to its space average at the given time T. Therefore we can find an equivalent of this form according to the position i:

$$\frac{1}{T} P_T = \frac{1}{T} \sum_{t=1}^{T} X_t \simeq_{T \to \infty} \sum_{i=-\infty}^{+\infty} \mathbb{P}(P_T = i) X_i$$

$\mathbb{P}(P_T = i)$ represents the probability for the walker to be at time T on pixel i. The transition random variable X_i depends on the position i only through its equivalence class $[i]$. Let $\lambda_{[i]}$ be the proportion of pixels in the equivalence class $[i]$ on the discrete line. Then we can write this sum in the form of:

$$\sum_{i=-\infty}^{+\infty} \mathbb{P}(P_T = i) X_i = \sum_{[i]} \left(\sum_{i \in [i]} \mathbb{P}(P_T = i) \right) X_{[i]} \simeq_{T \to \infty} \sum_{[i]} \lambda_{[i]} X_{[i]}$$

We build a sequence of random variables P_t with $\lambda_{[i]}$ proportion of pixels in the class $[i]$ on the discrete line, independent of the time. We have:

$$P_T \simeq T \sum_{[i]} \lambda_{[i]} X_{[i]}$$

We conclude by using the Central Limit Theorem, that $\frac{1}{\sqrt{T}} P_T$ follows a normal law. $\square$