Image Curvature Microscope

Jean-Michel Morel
Joint work with Adina Ciomaga and Pascal Monasse

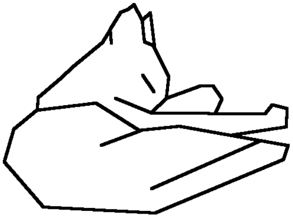
Centre de Mathématiques et de Leurs Applications,
Ecole Normale Supérieure de Cachan

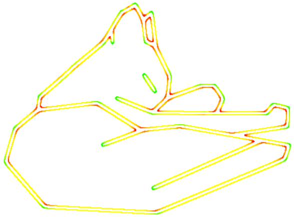
MVA Lecture Notes
October, 2010

Tout, au monde, existe pour aboutir un algorithme
(d'après Mallarmé)

Overview

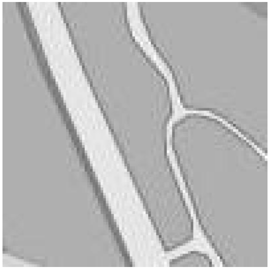
Role of Curvature in Visual Perception

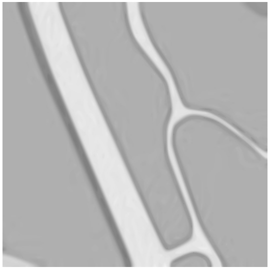

(a). Attneave's cat

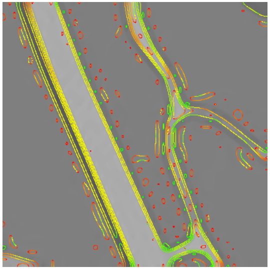

(b). Curvature map.

"Information is concentrated along contours and is further concentrated at those points on a contour at which its direction changes most rapidly" - *Attneave, '54*.

Aliasing and JPEG Artefacts


(a). Google Map.


(b). Filtering by LLS.

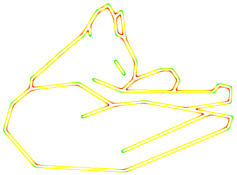

(c). Curvature Map.

Jordan Curve

Definition

We call *simple arc* or *Jordan arc* the image Γ of a continuous one-to-one function $\mathbf{x} : [0,1] \to \mathbb{R}^2$, $\mathbf{x}(t) = (x(t), y(t))$.

We say that Γ is a *simple closed curve* or *Jordan curve* if the mapping restricted to $(0,1)$ is one-to-one and if $\mathbf{x}(0) = \mathbf{x}(1)$.



Arc - length

Definition

If $\mathbf{x}$ is continuously differentiable on $[0, 1]$, we define the *arc length* of the segment of the curve between $\mathbf{x}(t_0)$ and $\mathbf{x}(t)$ by

$$L(\mathbf{x}, t_0, t) = \int_{t_0}^{t} |\mathbf{x}'(\tau)| \, \mathrm{d}\tau = \int_{t_0}^{t} \sqrt{\mathbf{x}'(\tau) \cdot \mathbf{x}'(\tau)} \, \mathrm{d}\tau. \tag{2.1}$$

In particular, set

$$L(t) = L(\mathbf{x}, 0, t) = \int_{0}^{t} |\mathbf{x}'(\tau)| \, \mathrm{d}\tau = \int_{0}^{t} \sqrt{\mathbf{x}'(\tau) \cdot \mathbf{x}'(\tau)} \, \mathrm{d}\tau.$$

Arc-length parameterization

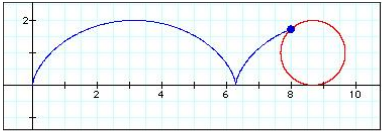
Definition

We say that a curve Γ admits an *arc-length parameterization* $s \mapsto \mathbf{x}(s)$ if the function $\mathbf{x}$ is C^1 and

$$L'(s) = |\mathbf{x}'(s)| = 1$$

for all s. In case Γ is closed, we identify $[0, l(\Gamma)]$ algebraically with the circle group by adding elements of $[0, l(\Gamma)]$ modulo $l(\Gamma)$.

We say that Γ is C^m, $m \in \mathbb{N}$, $m \geq 1$, if the arc-length parameterization $\mathbf{x}$ is a C^m function.



We want the definition of "smoothness" to describe an intrinsic property of Γ rather than a property of some parameterization $\mathbf{x}(s)$ of Γ. A C^∞ parameterized curve may not conform to our idea of being smooth, which at a minimum requires a tangent at every point $\mathbf{y} \in \Gamma$. For example, the motion of a point on the boundary of a unit disk as it rolls along the x-axis is described by $\mathbf{x}(t) = (t - \sin t, 1 - \cos t)$, which is a C^∞ function. Nevertheless, the curve has cusps at all multiples of 2π. The problem is that $\mathbf{x}'(2k\pi) = 0$.

Arc-length parameterization

Proposition

Suppose that Γ *is a* C^1 *Jordan curve with arc-length parameterization* $\mathbf{x} : [0, l(\Gamma)] \to \Gamma$*. Then any other arc-length parameterization* $\mathbf{y} : [0, l(\Gamma)] \to \Gamma$ *is of the form*

$$\mathbf{y}(s) = \mathbf{x}(s + \sigma)$$

or

$$\mathbf{y}(s) = \mathbf{x}(-s + \sigma)$$

for some $\sigma \in [0, l(\Gamma)]$*.*

Arc-length parameterization

Proof.

Denote by C the interval $[0, l(\Gamma)]$, defined as an additive subgroup of $\mathbb{R}$ modulo $l(\Gamma)$. Let $\mathbf{x}$, $\mathbf{y} : C \mapsto \Gamma$ be two length preserving parameterizations of Γ. Then

$$f = \mathbf{x} \circ \mathbf{y}^{-1}$$

is a length preserving bijection of C. Using the parameterization of C, this implies

$$f(s) = \pm s + \sigma \text{ for some } \sigma \in [0, l(\Gamma)]$$

and the proof is easily concluded. □

Tangent, normal and curvature vectors

Definition

Assume that Γ is C^2 and let $s \mapsto \mathbf{x}(s)$ be an arc-length parameterization. The *tangent vector* $\boldsymbol{\tau}$ is defined as

$$\boldsymbol{\tau}(s) = \mathbf{x}'(s).$$

The *curvature vector* of the curve Γ is defined by

$$\boldsymbol{\kappa}(s) = \mathbf{x}''(s).$$

The *normal vector* $\mathbf{n}(s)$ is defined by

$$\mathbf{n}(s) = \boldsymbol{\tau}^{\perp},$$

where $(x, y)^{\perp} = (-y, x)$.

Tangent, normal and curvature vectors

Proposition

Let Γ be a C^2 Jordan curve, and let $\mathbf{x}$ and $\mathbf{y}$ by any two arc-length parameterizations of Γ.

(i) *If $\mathbf{x}(s) = \mathbf{y}(t)$, then $\mathbf{x}'(s) = \pm\mathbf{y}'(t)$.*

(ii) *The vector $\boldsymbol{\kappa}$ is independent of the choice of arc-length parameterizations and it is orthogonal to $\boldsymbol{\tau} = \mathbf{x}'$.*

Tangent, normal and curvature vectors

Proof.

From the previous theorem we have $\mathbf{y}(s) = \mathbf{x}(\pm s + \sigma)$ and (i) follows by differentiation. This is also geometrically obvious: $\mathbf{x}'(s)$ and $\mathbf{y}'(t)$ are unit vectors tangent to Γ at the same point. Thus, they either point in the same direction or they point in opposite directions.

Using any of the above representations and differentiating twice

$$\mathbf{x}'' = \mathbf{y}''.$$

Since $\mathbf{x}' \cdot \mathbf{x}' = 1$, differentiating this expression shows that

$$\mathbf{x}'' \cdot \mathbf{x}' = 0.$$

Thus, $\mathbf{x}''$ and $\mathbf{x}'$ are orthogonal and $\mathbf{x}''$ and $\mathbf{x}'^{\perp}$ are collinear. □

Curvature vector

Definition (and notation)

Given a C^2 curve Γ, which is parameterized by length as $s \mapsto \mathbf{x}(s)$ and $\mathbf{x} = \mathbf{x}(s)$ a point of Γ, we denote in three equivalent ways the curvature of Γ at $\mathbf{x} = \mathbf{x}(s)$,

$$\boldsymbol{\kappa}(\mathbf{x}) = \boldsymbol{\kappa}(\mathbf{x}(s)) = \boldsymbol{\kappa}(s) = \mathbf{x}''(s).$$

In the first notation, $\boldsymbol{\kappa}$ is the curvature of the curve Γ at a point $\mathbf{x}$ implicitly supposed to belong Γ. In the second notation a particular parameterization of Γ, $\mathbf{x}(s)$, is being used. In the third one, $\mathbf{x}$ is omitted.

Notations and conventions

- No distinction between a Jordan curve Γ as a subset of the plane and a function $s \mapsto \mathbf{x}(s)$ such that $\Gamma = \{\mathbf{x}(s)\}$.
- Families of Jordan curves dependent on a parameter $t > 0$, we will most often denote these families by $\mathbf{x}(t, s)$,
- $\mathbf{x}(t, s)$ has three meanings: a family of Jordan curves, a family of functions that represent these curves, and a particular point on one of these curves.

Curve evolution

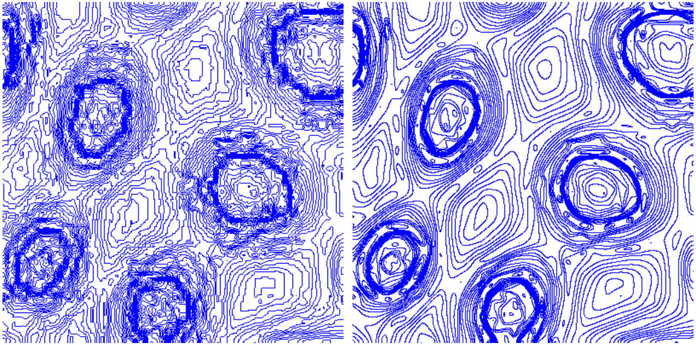


Figure: Simultaneous curve smoothing.

Intrinsic Heat Equation or Curve Shortening

Definition

Let $\mathbf{x}(t)$, $t > 0$, be a family of C^2 Jordan curves. We say that $\mathbf{x}(t)$ satisfies the intrinsic heat equation if

$$\frac{\partial \mathbf{x}}{\partial t} = \boldsymbol{\kappa}(\mathbf{x}(t)). \tag{CS}$$

Intrinsic Heat Equation or Curve Shortening

Theorem (Grayson)

Let $\mathbf{x}_0$ be a C^1 Jordan curve. By using the intrinsic heat equation, it is possible to evolve $\mathbf{x}_0$ into a family of Jordan curves $\mathbf{x}(t,s)$ such that $\mathbf{x}(0,s) = \mathbf{x}_0(s)$ and such that for every $t > 0$, $\mathbf{x}(t,s)$ is C^∞ (actually analytical) and satisfies the equation (CS).

Furthermore, for every $t > 0$, $\mathbf{x}(t,s)$ has only a finite number of inflection points and curvature extrema, and the number of these points does not increase with t. For every initial curve, there is a scale t_0 such that the curve $\mathbf{x}(t,s)$ is convex for $t \geq t_0$ and there is a scale t_1 such that the curve $\mathbf{x}(t,s)$ is a single point for $t \geq t_1$. This evolution satisfies the **inclusion principle: if $\mathbf{x}_0$ surrounds $\mathbf{y}_0$ then $\mathbf{x}(t)$ surrounds $\mathbf{y}(t)$.**

Affine Shortening

A surprising variant of Curve Shortening is given by the Affine Shortening equation

$$\frac{\partial \mathbf{x}}{\partial t} = |\boldsymbol{\kappa}|^{-\frac{2}{3}} \boldsymbol{\kappa}(\mathbf{x}(t)) \tag{AS}$$

Theorem (Angenent, Sapiro, Tannenbaum)

Let $\mathbf{x}_0$ be a C^2 Jordan curve. Then there is a unique classical solution $\mathbf{x}(t)$ of (AS). The curve eventually becomes convex and thereafter evolves towards an ellipse before collapsing.

Dynamic Shape Evolution

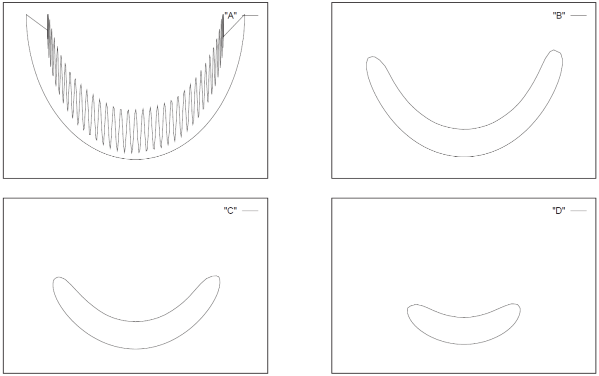


Figure: Curve evolution by the intrinsic heat equation.

Dynamic Shape Evolution

Algorithm (Mackworth and Mokhtarian)

```
Input: Polygon $\Gamma_0$, gaussian-like kernel $G$
Output: Evolved polygon $\Gamma_n$, after $n$ iterations
1 for all $i = \overline{0, n}$ do
2     sample uniformly curve $\Gamma_i$;
3     convolve curve $\Gamma_i$ with $G$.
```

Discrete Curve Shortening

Theorem

Let $\mathbf{x}$ *be a* C^2 *curve parameterized by its length parameter* $s \in [0, L]$ *and* G *a centered probability density law with variance* $2c$. *Set* $G_h(x) = \frac{1}{h^{\frac{1}{2}}} G(\frac{x}{h^{\frac{1}{2}}})$. *Then*

$$G_h * \mathbf{x}(s) - \mathbf{x}(s) = ch\boldsymbol{\kappa}(\mathbf{x}(s)) + o(h). \tag{2.2}$$

where c *is a positive constant.*

Dynamic Shape Evolution

Proof.

Using the definition of the convolution we write the left-hand side of the equation as

$$\begin{aligned} G_h * \mathbf{x}(s) - \mathbf{x}(s) &= \int G_h(s-\tau)\mathbf{x}(\tau)d\tau - \int G_h(\tau)\mathbf{x}(s)d\tau \\ &= \int \frac{1}{h^{1/2}} G(\frac{\tau}{h^{1/2}})[\mathbf{x}(s-\tau) - \mathbf{x}(s)]d\tau \\ &= \int G(\tau)[\mathbf{x}(s - h^{1/2}\tau) - \mathbf{x}(s)]. \end{aligned}$$

□

Dynamic Shape Evolution

Proof.

By Taylor's formula with Peano remainder

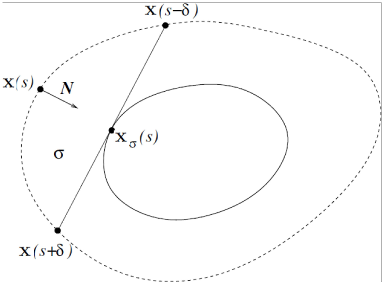
$$\mathbf{x}(s-h^{1/2}\tau)-\mathbf{x}(s)=-h^{1/2}\tau\frac{\partial\mathbf{x}}{\partial s}(s)+\frac{h}{2}\tau^2\frac{\partial^2\mathbf{x}}{\partial s^2}(s)+\frac{h}{2}\alpha(s-h^{1/2}\tau),$$

with α a continuous function such that $\lim_{h\to 0}\alpha(s-h^{1/2}\tau)=0$. Thus, we have

$$\begin{aligned} G_h*\mathbf{x}(s)-\mathbf{x}(s) &= h\frac{\partial^2\mathbf{x}}{\partial s^2}(s)\frac{1}{2}\int G(\tau)\tau^2+\frac{h}{2}\int\alpha(s-h^{1/2}\tau)d\tau \\ &= ch\boldsymbol{\kappa}(\mathbf{x}(s))+o(h). \end{aligned}$$

$\square$

Discrete Affine Shortening



Definition

A σ-chord is a chord determined by two points of the curve $(\mathbf{x}(s-\delta), \mathbf{x}(s+\delta))$, where $\delta > 0$ is chosen in order that the area of the region enclosed by this chord and the piece of curve $\mathbf{x}\big|_{(s-\delta, s+\delta)}$ be equal to σ.

Discrete Affine Shortening

Algorithm (Moisan)

```
Input: Polygon $\Gamma_0$,
Output: Evolved polygon $\Gamma_\sigma$, at scale $\sigma^{2/3}$
4 break the curve into convex and concave parts  for every convex/concave component do
5     replace each component by the sequence of the middle points of each $\sigma$-chord such that one endpoint is a vertex of the polygonal curve;
6     concatenate the pieces of curves previously obtained.
```

Discrete Affine Shortening

Theorem

Let $\mathbf{x}$ be a C^2 curve parameterized by its length parameter $s \in [0, L]$ and $\sigma > 0$. To each point of $\mathbf{x}(s)$, we associate $\mathbf{x}_\sigma(s)$, defined as the middle point of the σ-chord $(\mathbf{x}(s-\delta), \mathbf{x}(s+\delta))$, where $\delta > 0$ is chosen in order that the area of the region enclosed by this chord and the piece of curve $\mathbf{x}\big|_{(s-\delta, s+\delta)}$ be equal to σ. Then

$$\mathbf{x}_\sigma(s) - \mathbf{x}(s) = c\sigma^{2/3}|\boldsymbol{\kappa}(\mathbf{x})|^{-2/3}\boldsymbol{\kappa}(\mathbf{x}) + o(\sigma^{2/3}) \text{ as } \sigma \to 0$$

where c is a positive constant.

From Images to Curves: Implicit Function Theorem

Theorem

Let $u \in \mathcal{F}$ be a C^1 function such that $Du(\mathbf{x}_0) \neq 0$ at some $\mathbf{x}_0 = (x_0, y_0)$. Let $\mathbf{i}$ denote the unit vector in the direction (u_x, u_y), let $\mathbf{j}$ denote the unit vector in the orthogonal direction $(-u_y, u_x)$, and write $\mathbf{x} = \mathbf{x}_0 + x\mathbf{i} + y\mathbf{j}$. Then there is a disk $D(\mathbf{x}_0, r)$ and a unique C^1 function φ, $\varphi : [-r, r] \to \mathbb{R}$, such that if $\mathbf{x} \in D(\mathbf{x}_0, r)$, then

$$u(x, y) = u(\mathbf{x}_0) \quad \Longleftrightarrow \quad x = \varphi(y).$$

From Images to Curves: Structure of Level Lines

Corollary

Assume that $u \in \mathcal{F}$ *is* C^1 *and let* $u^{-1}(\lambda) = \{\mathbf{x} \mid u(\mathbf{x}) = \lambda\}$ *for* $\lambda \in \mathbb{R}$*. If* $\lambda \neq u(\infty)$ *and* $Du(\mathbf{x}) \neq 0$ *for all* $\mathbf{x} \in u^{-1}(\lambda)$*, then* $u^{-1}(\lambda)$ *is a finite union of disjoint Jordan curves.*

Proof.

From the Implicit Function Theorem we know that for each point $\mathbf{x} \in u^{-1}(\lambda)$ there is an open disk $D(\mathbf{x}, r(\mathbf{x}))$ such that $\overline{D}(\mathbf{x}, r(\mathbf{x})) \cap u^{-1}(\lambda)$ is a C^1 Jordan arc $\mathbf{x}(s)$ and we can take the endpoints of the arc on $\partial D(\mathbf{x}, r(\mathbf{x}))$. Since $\lambda \neq u(\infty)$, $u^{-1}(\lambda)$ is compact. Thus there is a finite number of points $\mathbf{x}_i$, $i = 1, \ldots, m$, such that

$$u^{-1}(\lambda) \subset \bigcup_{i=1}^{m} D(\mathbf{x}_i, r(\mathbf{x}_i)).$$

This implies that $u^{-1}(\lambda)$ is a finite union of Jordan arcs which we can parameterize by length. The rest of the proof is very intuitive and is left to the reader. It consists of iteratively gluing the Jordan arcs until they close up into one or several Jordan curves. $\square$

From Images to Curves: Level Lines Structure

Theorem (Sard's theorem)

Let $u \in \mathcal{F} \cap C^1$. Then for almost every λ, the set $u^{-1}(\lambda)$ is nonsingular, which means that for all $\mathbf{x} \in u^{-1}(\lambda)$, $Du(\mathbf{x}) \neq 0$.

Corollary

Let $u \in \mathcal{F} \cap C^1$. Then for almost every λ in the range of u, the set $u^{-1}(\lambda)$ is the union of a finite set of disjoint Jordan C^1 curves.

Proposition

Let $u \in \mathcal{F} \cap C^1$. Then u can be reconstructed from the following data: the family of all of its level lines at nonsingular levels, the level of each level line being also kept.

From Images to Curves: Level Lines Structure

Proof.

Let G be the closure of the union of the ranges of all level lines of u at nonsingular levels. If $\mathbf{x} \in G$, then there are points $\mathbf{x}_n$ belonging to level lines of some levels λ_n such that $\mathbf{x}_n \to \mathbf{x}$. Hence, $\lambda_n = u(\mathbf{x}_n) \to u(\mathbf{x})$. So we get back the value of $u(\mathbf{x})$.

Let now $\mathbf{x}$ belong to the open set G^c. Let us first prove that $Du(\mathbf{x}) = 0$. Assume by contradiction that $Du(\mathbf{x}) \neq 0$. By using the first order Taylor expansion of u around $\mathbf{x}$, one sees that for all $r > 0$ the connected range $u(B(\mathbf{x}, r))$ must contain some interval $(u(\mathbf{x}) - \alpha(r), u(\mathbf{x}) + \alpha(r))$ with $\alpha(r) \to 0$ as $r \to 0$. □

From Images to Curves: Level Lines Structure

Proof.

(continues) By Sard's theorem some of the values in this interval are nonsingular. Thus we can find nonsingular levels $\lambda_n \to u(\mathbf{x})$ and points $\mathbf{x}_n \to \mathbf{x}$ such that $u(\mathbf{x}_n) = \lambda_n$. This implies that $\mathbf{x} \in G$ and yields a contradiction.

Thus $Du(\mathbf{x}) = 0$ and u is therefore constant on each connected component A of G^c. The value of u is then uniquely determined by the value of u on the boundary of A. This value is known, since ∂A is contained in G. $\square$

Pixelization effects

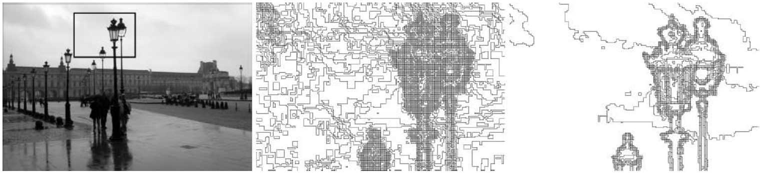


Figure: Level lines from a digital image (zero-order interpolation). Left: original image. Middle: all level lines for the small image inside the rectangle. Right: level lines for a gray level quantization step of 10.

Pixelization effects

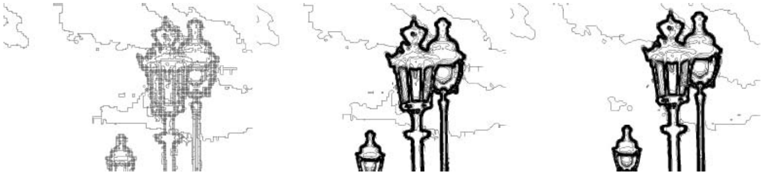
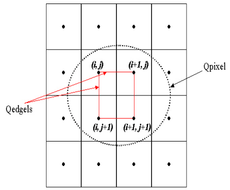


Figure: Left: level lines from a digital image (zero-order interpolation). The quantization step for the gray levels is 10, starting from 10. Middle: level lines from the piecewise bilinear interpolated image. The quantization step for the gray levels is 10, starting from 10. Right: level lines from the piecewise bilinear interpolated image, with a gray level quantization step of 10, starting at gray level 0:5.

Bilinear interpolation

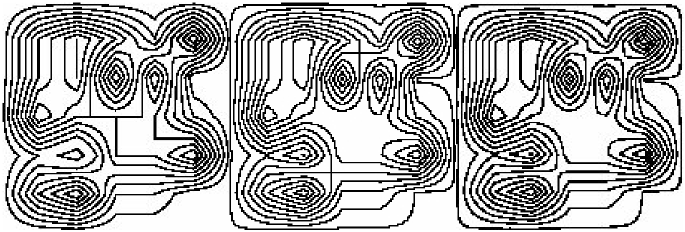


The general form of a bilinear function is

$$u(x,y) = axy + bx + cy + d.$$

For each set of four adjacent pixels (*Qpixel*) parameters a, b, c and d are fixed by the gray levels of the four pixels .

Bilinear Level Lines



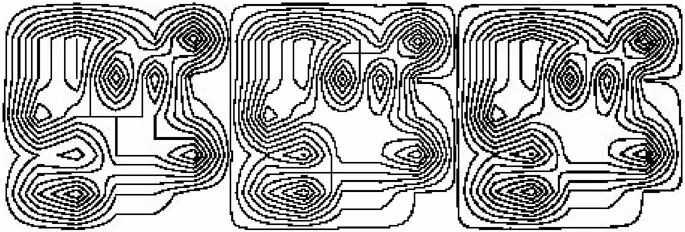
The equation for a level line at level λ of the bilinear interpolated image inside a dual pixel can be written

$$a(x - x_s)(y - y_s) + (\lambda_s - \lambda) = 0$$

or

$$axy + bx + cy + (d - \lambda) = 0.$$

Bilinear Level Lines



Proposition

(Properties of level lines of bilinear interpolates)

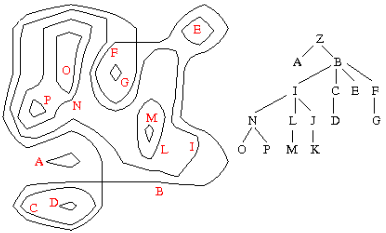
- *Except for a finite set $\lambda_1, \dots, \lambda_n$ of levels, the level set $u(x, y) = \lambda$ is the union of a finite number of Jordan curves;*
- *All level lines are continuous and piecewise C^1.*

Bilinear tree of level lines

- One can decompose an image into a finite set of *tagged* polygonal lines, indexed by half-integer gray values

$$\mathcal{T}_0 = \{\Gamma_0^{\lambda,i}; i \in F^\lambda, \lambda \in \mathbb{N} + 1/2\}. \qquad (3.3)$$

- The set is ordered in a tree structure, induced by the geometrical inclusion.



Direct algorithm: level lines decomposition

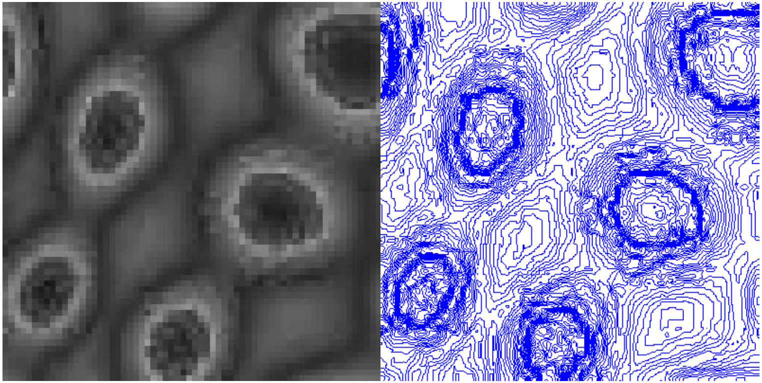


Figure: Reconstruction of the image from a family of level lines.

Direct algorithm: level lines decomposition

Algorithm

```
**Input**: List $L$ of bilinear level lines $\Gamma$
**Output**: Fill tree structure
7  Collect all $(i, y, id)$ in array $V$;
8  /* Intersection of curve index id with Qedgel at column $i + 0.5$ */;
9  Order $V$ lexicographically by key $(i, y)$;
10 $\Gamma \leftarrow \emptyset$; /* Innermost shape */;
11 for all $(i, y, id) \in V$ do if $\Gamma = \Gamma_{id}$ then
12     $\Gamma \leftarrow Parent(\Gamma)$; /* Getting out of innermost shape */;
13     else if $\Gamma \neq \emptyset$ and $Parent(\Gamma_{id}) = \emptyset$ then
14         set $\Gamma_{id}$ as a child of $\Gamma$;
15         $\Gamma \leftarrow \Gamma_{id}$. /* Innermost shape is now $\Gamma_{id}$ */
```

Inverse algorithm: image reconstruction

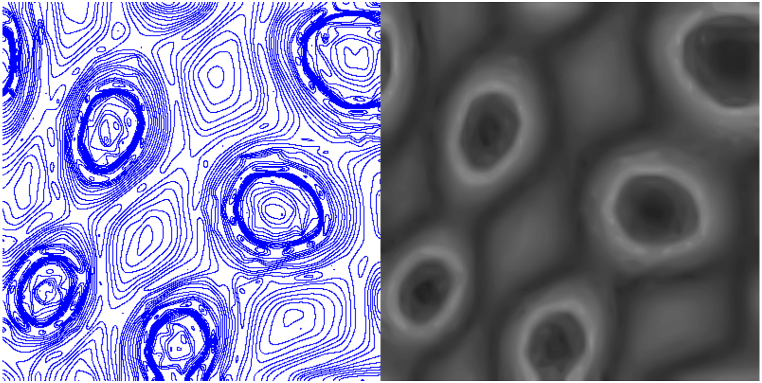


Figure: Reconstruction of the image from a family of level lines.

Inverse algorithm: image reconstruction

The algorithm described in this section performs an exact image reconstruction from a topographic map, i.e. from an arbitrary family of Jordan curves organized in a tree structure with respect to geometrical inclusion.

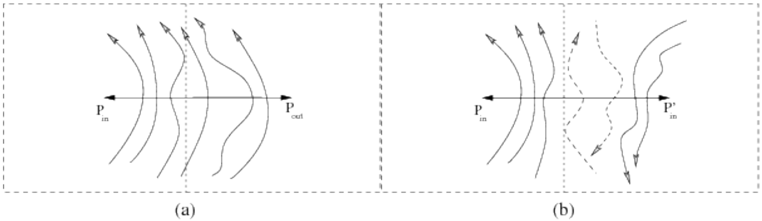


Figure: The level line $2D$ inclusion topology is reflected in the $1D$ ordering of their intersections with the dual edges.

Inverse algorithm: image reconstruction

A closed curve Γ is stored as a set of ordered points $\{P_k(x_k, y_k)\}_{1 \leq k \leq N}$ with N depending on Γ. The real numbers x_k and y_k are the floating point coordinates of the vertex number k of the polygon Γ. We need to fill in all pixels with integral coordinates (j, i) inside the polygon.

Polygon intersections with the grid

The goal of this Algorithm, which is a preliminary to the filling algorithm, is to find the intersections of the polygonal level line with all horizontal lines $y = i$. For any given i the intersection is in fact the intersection of a segment $[P_k P_{k+1}]$ of the polygon with the line $y = i$. These intersections are ordered by their abscissae so that

$$x_1^i \leq x_2^i \leq \cdots \leq x_p^i,$$

where p is even because Γ is a closed curve. This gives a simple and fast decision rule: a pixel (j, i) is surrounded by the polygon if and only if j is within an odd interval $[x_{2k+1}^i, x_{2k+2}^i]$.

Polygon intersections with the grid

Algorithm

Input: *Vertices $P_k(x_k, y_k)$ of polygon Γ*

Output: *For each i, the ordered list L^i of points of Γ on the line of equation $y = i$*

```
16 for all $i$ do $L^i \leftarrow \emptyset$   for all segments $[P_k P_{k+1}]$ do
17     for $i \in [y_k, y_{k+1}] \cap \mathbb{N}$ do
18         $(x, i) \leftarrow [P_k P_{k+1}] \cap \{y = i\}$  Insert $x$ in $L^i$
19 for all $i$ do sort list $L^i$
```

Filling the interior

Line by line all odd intervals on L^i are enumerated and filled in with level $\lambda \pm 1/2$ at all pixels with ordinate i whose abscissa is inside such an interval:

Algorithm

Input: *Sorted lists L^i of intersections of Γ with lines $\{y = i\}$, level λ*

Output: *Pixels inside polygon Γ are at level $\lambda \pm 1/2$, pixels outside unchanged*

```
20 for all $i$ do
21     for all $x^i_{2k+1} \in L^i$ do
22         for $j \in \mathbb{N} \cap [x^i_{2k+1}, x^i_{2k+2}]$ do
23             pixel $(j, i) \leftarrow \lambda \pm 1/2$
```

Curvatures computed directly on level lines

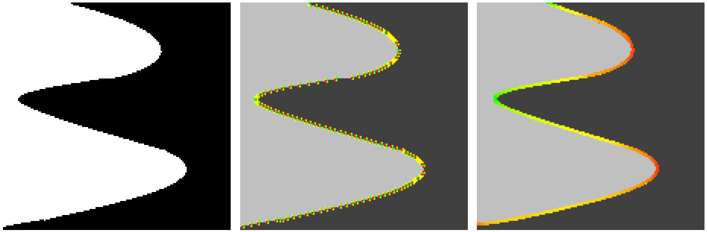


Figure: The curvature color display rule. Initial image, FDS and LLS.

Discrete curvature for a polygonal line.

We recall that each level line is stored as a set of ordered points

$$\Gamma = \{P_i(x_i, y_i)\}_{i=0..n}, \text{ with } P_0 = P_n.$$

The discrete scalar curvature k_i computed at each vertex P_i is obtained as the inverse of the circumscribed radius R_i of the triangle $P_{i-1}P_iP_{i+1}$.

Lemma

The curvature at vertex P_i is given by

$$k_i = 2\frac{u_i^1 u_{i+1}^2 - u_i^2 u_{i+1}^1}{u_i u_{i+1} v_i}. \tag{4.4}$$

Subpixel curvature algorithm

Algorithm (Ciomaga, Monasse, Morel, '10)

```
Input: Original Image $u_0$.
Output: Curvatures $u_0$ at scale $t$: $u(\cdot, t)$.
24 Extract the tree of level lines $\{\Gamma_0^{\lambda,i}\}_{i \in F_\lambda; \lambda}$;
25 Sample uniformly each level line $\Gamma_0^{\lambda,i}$
26 for Level line $\Gamma_0^{\lambda,i}$ do
27     $\Gamma_t^{\lambda,i}$ = Curve Shortening Flow ($\Gamma_0^{\lambda,i}$);
28 for $\Gamma_t^{\lambda,i} = \{P_i(x_i, y_i)\}_{i=0..n}$ do
29     $k_i = 1/R_i$;
30 for each dual pixel do
31     $k = mean(k_{i_1}, k_{i_2}, ..., k_{i_m})$.
```

Signed and topological curvatures

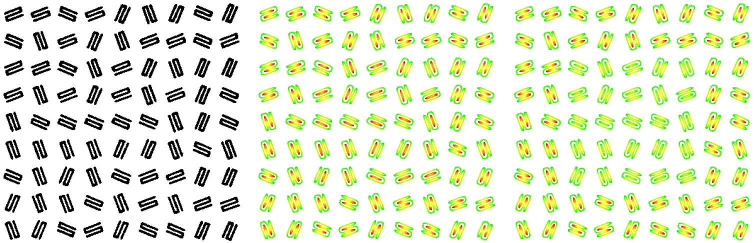


Figure: Original image, signed curvatures and topological curvatures

Curvature Microscope

Figure: Original image, 2X zoom and 4X zoom of the up-right corner. A zoom is necessary to observe the single curvatures.

Curvature Microscope

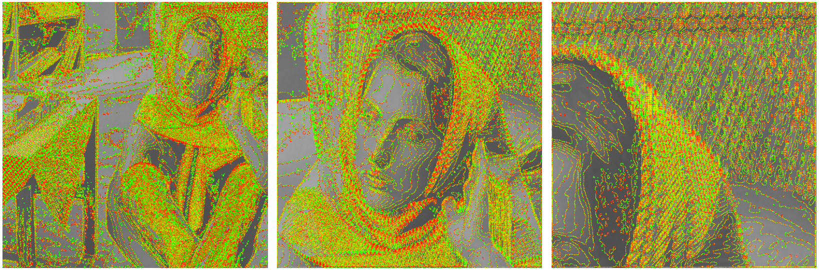


Figure: Curvature map computed on the original level lines with a quantization step $s = 16$.

Curvature Microscope

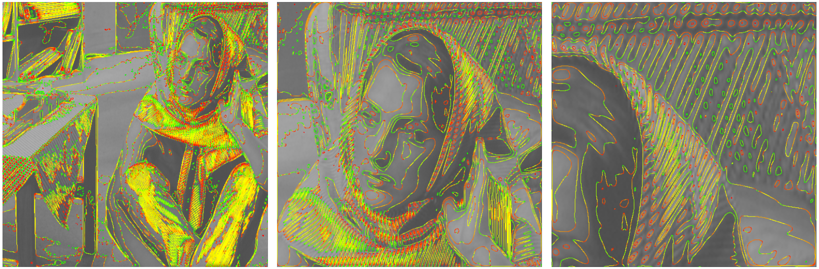


Figure: Curvature map computed on shortened level lines at normalized scales $l = 1$, $l = 2$, and $l = 4$.

A closer look at Attneave's cat

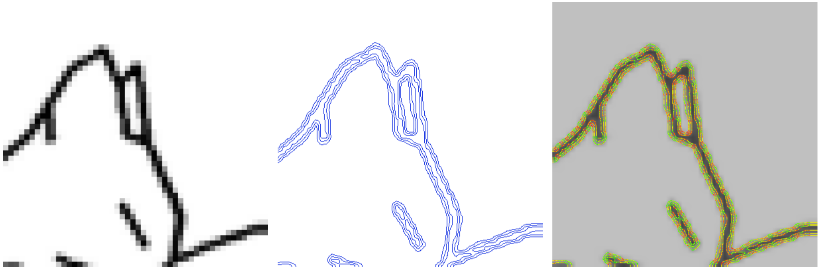


Figure: Zoom on the Attneave cat, its corresponding level lines and curvatures.

A closer look at Attneave's cat

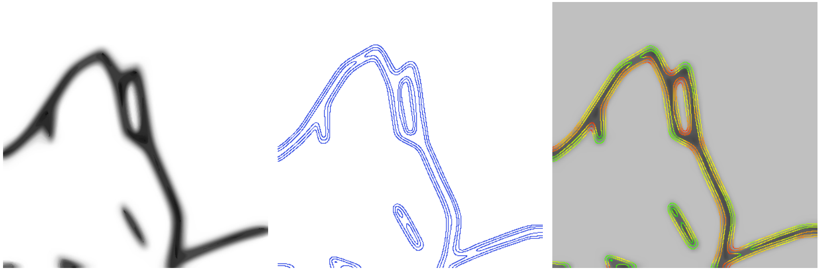


Figure: LLAS evolution, affine smoothed level lines and curvature map after filtering.

Graphics and aliasing

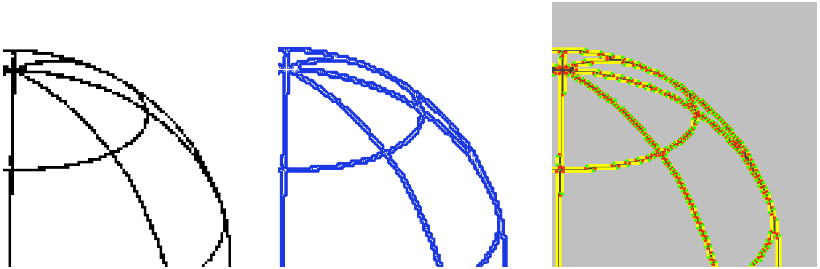


Figure: Original image, its corresponding level lines and curvatures.

Graphics and aliasing

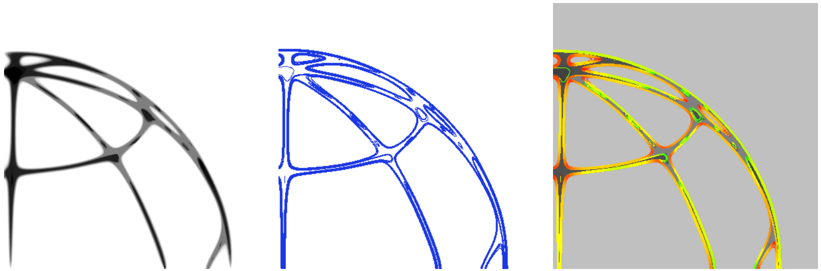
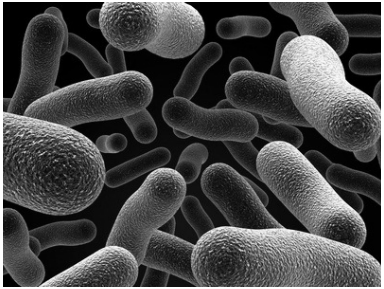


Figure: LLAS evolution, affine smoothed level lines and curvature map after filtering.

Bacteria morphologies



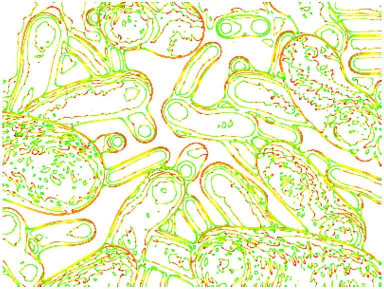


Figure: Original bacteria image and the corresponding curvature map.

Digital elevation models

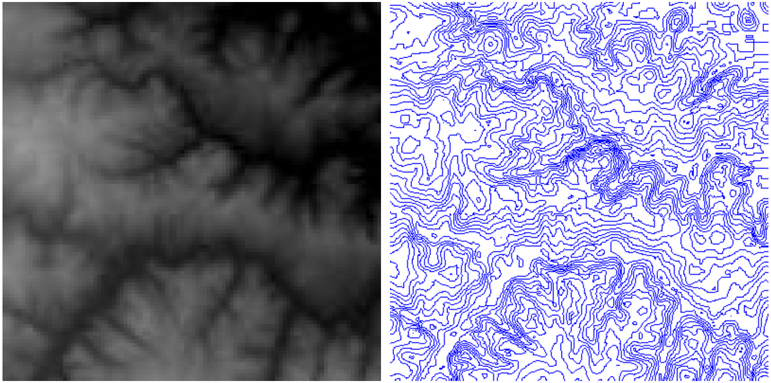


Figure: Digital elevation map and its corresponding level lines.

Digital elevation models

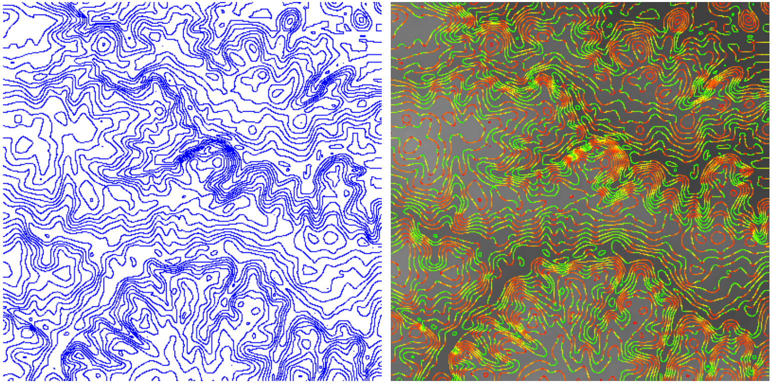


Figure: The affine smoothed level lines and their curvature map.

JPEG artefact reduction

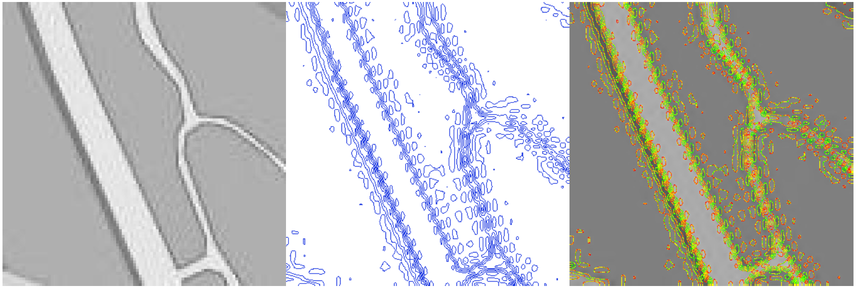


Figure: Piece of map with roads, its corresponding level lines and curvatures.

JPEG artefact reduction

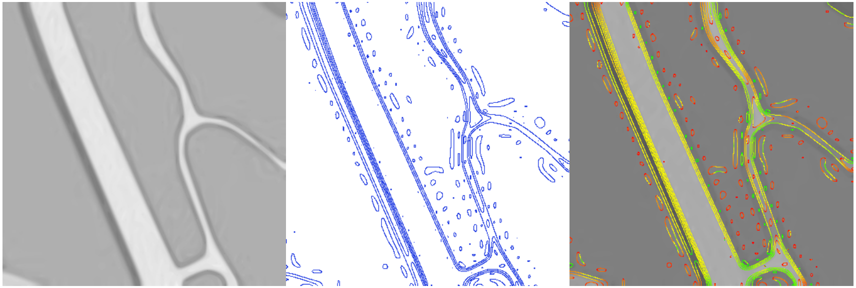


Figure: LLAS evolution, affine smoothed level lines and curvature map after filtering.

Paitings *sfumato technique*

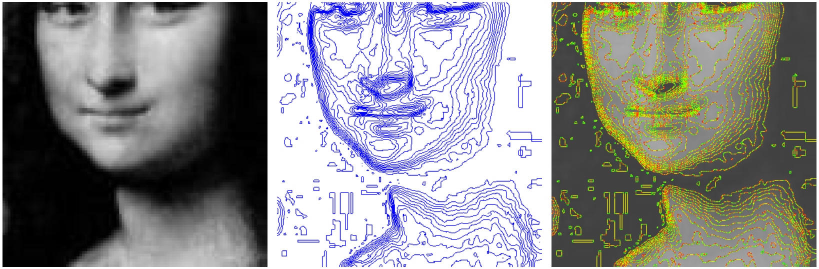


Figure: Extraction with zoom of *Mona Lisa* photograph, its corresponding level lines and curvatures.

Paitings *sfumato technique*

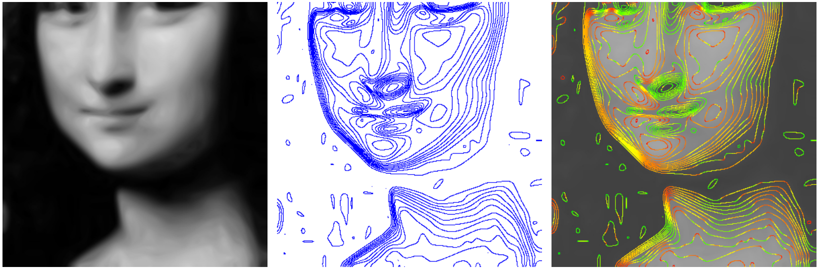


Figure: LLAS evolution, affine smoothed level lines and curvature map after filtering.

Text processing

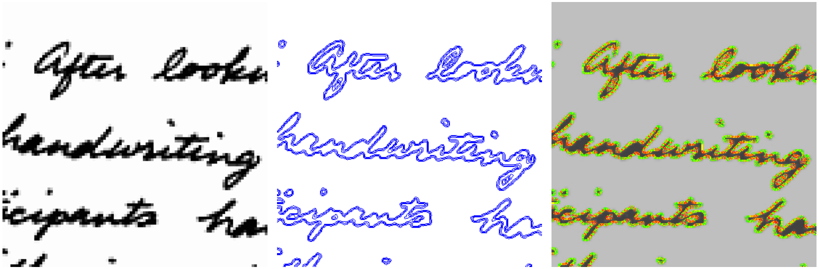


Figure: Original handwriting, corresponding level lines and curvatures.

Text processing

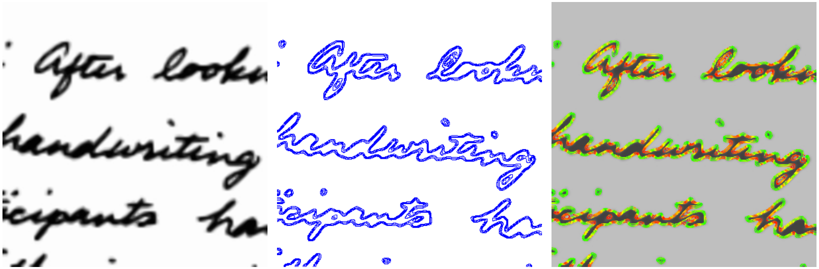


Figure: LLAS evolution, affine smoothed level lines and curvature map after filtering.

Fingerprints restoration and discrimination

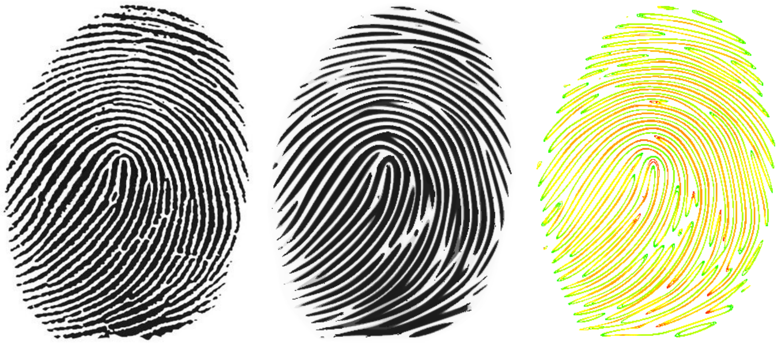


Figure: Original fingerprint, Level Lines Affine Shortening and its Curvature map.

Conclusion

- The first outcome of the Level lines Shortening algorithm is the evolved image, which presents some sort of denoising, simplification, and desaliasing;
- The main outcome is an accurate curvature estimate on all level lines;
- A powerful visualization tool, due to the fact that all level lines are polygons with real coordinates allows to zoom in the image at an arbitrary resolution;