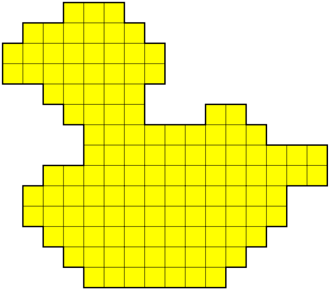
Dynamic Minimum Length Polygon

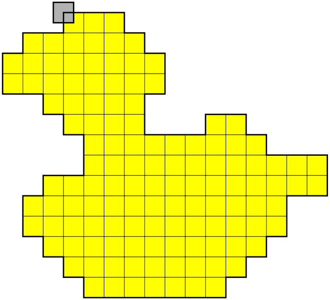
J.-O. Lachaud, X. Provençal

3 mars 2015

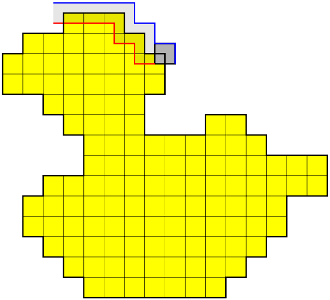
Minimum Length Polygon



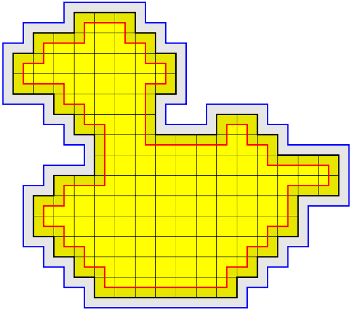
Minimum Length Polygon



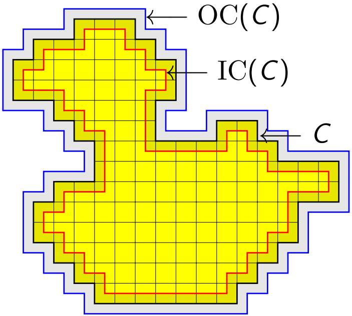
Minimum Length Polygon



Minimum Length Polygon



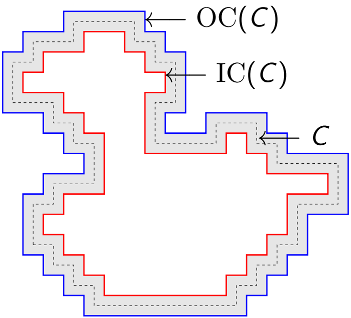
Minimum Length Polygon



Definition

Given a digital contour C, its *inner* (resp. *outer*) *contour* $\mathrm{IC}(C)$ (resp. $\mathrm{OC}(C)$) is the erosion (resp. dilatation) of the body of $I(C)$ by the open unit square centrer on $(0,0)$.

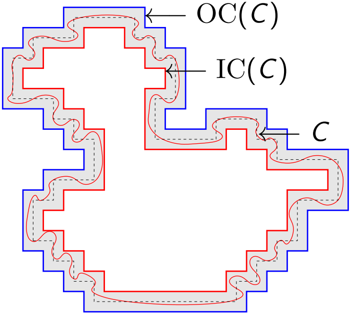
Minimum Length Polygon



Definition

Given a digital contour C, its *inner* (resp. *outer*) *contour* $\mathrm{IC}(C)$ (resp. $\mathrm{OC}(C)$) is the erosion (resp. dilatation) of the body of $I(C)$ by the open unit square centrer on $(0, 0)$.

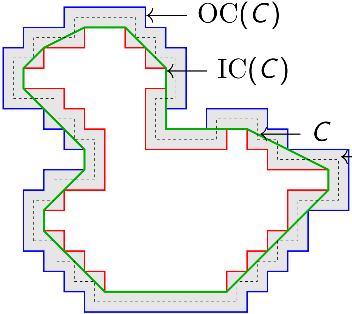
Minimum Length Polygon



Definition

Given a digital contour C, its *inner* (resp. *outer*) *contour* $\mathrm{IC}(C)$ (resp. $\mathrm{OC}(C)$) is the erosion (resp. dilatation) of the body of $I(C)$ by the open unit square centrer on $(0, 0)$.

Minimum Length Polygon



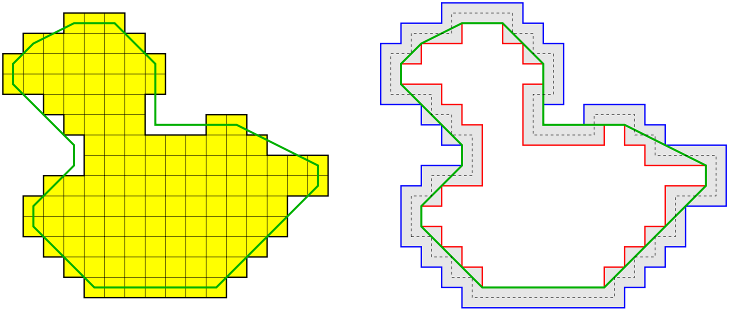
Definition

The *minimum length polygon* of C is a subset $P \in \mathbb{R}^2$ such that,

$$P = \underset{A \in \mathcal{A},\, \mathrm{IC}(C) \subseteq A,\; \partial A \subset \mathrm{OC}(C) \setminus \mathrm{IC}(C)^\circ}{\arg\min} \mathrm{Per}(A)$$

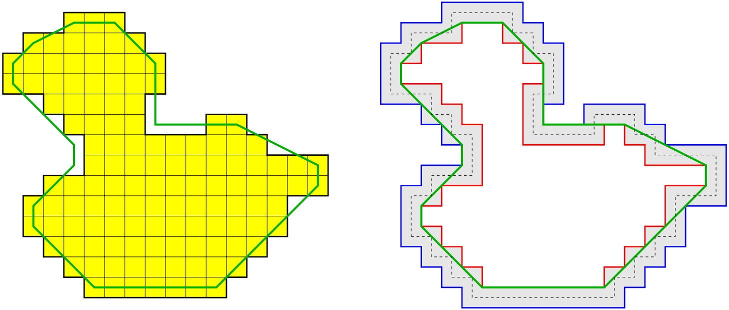
where $\mathcal{A}$ is the family of simply connected compact sets of $\mathbb{R}^2$.

Minimum Length Polygon



The MLP is a polygonal line whose vertices are centers of pixels along the inner or the outer contour, also :

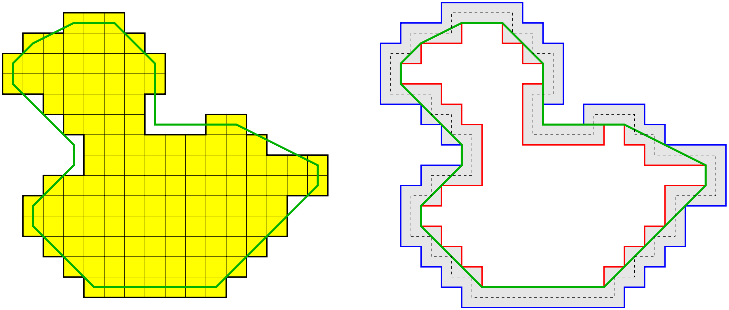
Minimum Length Polygon



The MLP is a polygonal line whose vertices are centers of pixels along the inner or the outer contour, also :

- unique ;

Minimum Length Polygon

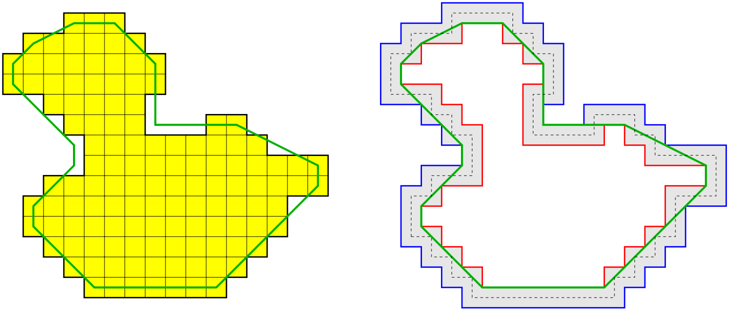


The MLP is a polygonal line whose vertices are centers of pixels along the inner or the outer contour, also :

- unique ;
- a good length estimator[1] ;

[1] Proved to be convergent on convex shapes.

Minimum Length Polygon

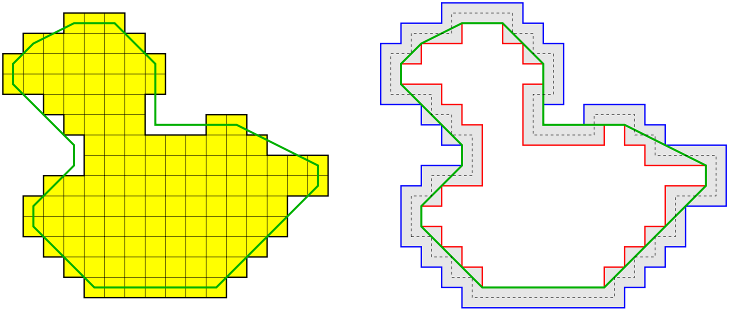


The MLP is a polygonal line whose vertices are centers of pixels along the inner or the outer contour, also :

- unique ;
- a good length estimator[1] ;
- a good tangent estimator ;

[1] Proved to be convergent on convex shapes.

Minimum Length Polygon

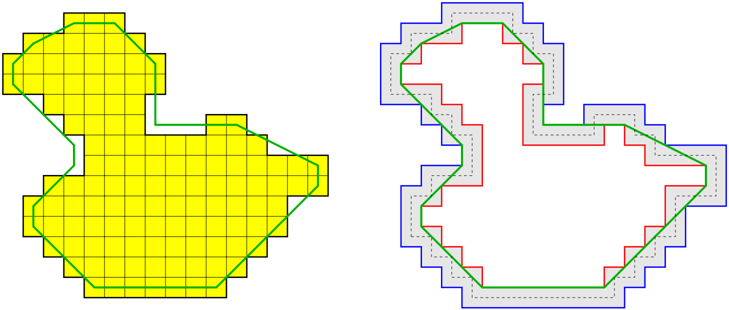


The MLP is a polygonal line whose vertices are centers of pixels along the inner or the outer contour, also :

- unique ;
- a good length estimator[1] ;
- a good tangent estimator ;
- characteristic of the shape's convexity ;

[1] Proved to be convergent on convex shapes.

Minimum Length Polygon



The MLP is a polygonal line whose vertices are centers of pixels along the inner or the outer contour, also :

- unique ;
- a good length estimator[1] ;
- a good tangent estimator ;
- characteristic of the shape's convexity ;
- reversible[2].

[1] Proved to be convergent on convex shapes.
[2] If aligned vertices are considered.

Computation of MLP

MLP is computable in time linear with respect of the length of C.

- J.-O. Lachaud, X. Provençal, *Two linear-time algorithms for computing the minimum length polygon of a digital contour*, Discrete Applied Mathematics (DAM), 2011.

Segmentation using deformable models

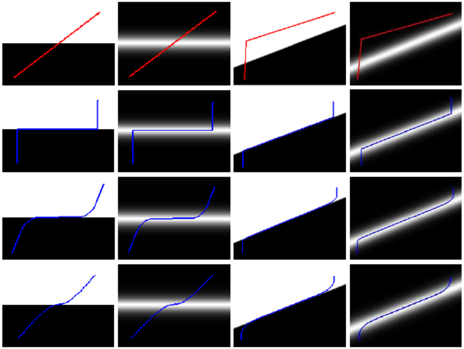
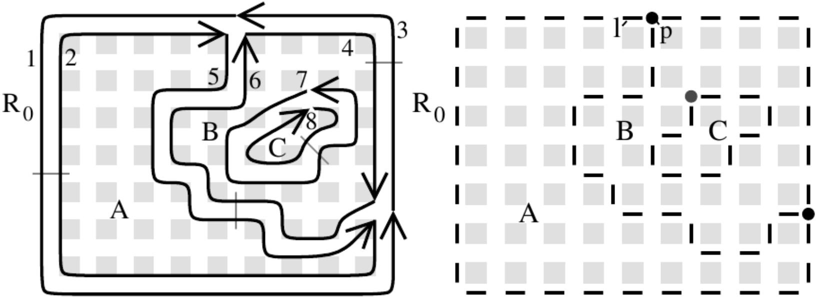


Fig. 4. Example of the minimization process using the Greedy1 algorithm. The gradient is computed with the Canny-Deriche method with scale coefficient 0.2. The input image represents a half-plane. (First row) Initialisation of the DDM. (Second row) Results of the minimisation process, the α coefficient used is equal to 0. (Third row) Results with $\alpha = 200$. (Fifth row) Results with $\alpha = 300$.

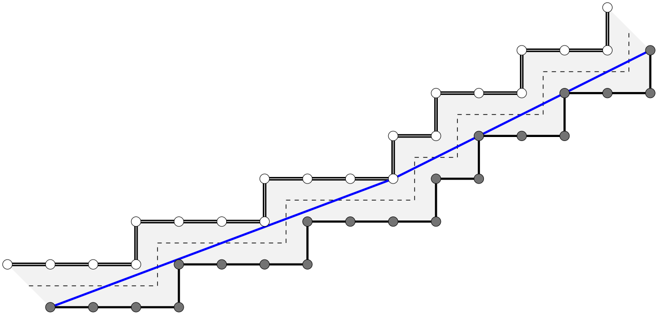
- F. de Vieilleville and J.-O. Lachaud, *Digital Deformable Model Simulating Active Contours*, in proc. DGCI2009, LNCS 5810, p.203-216, 2009.

Segmentation using deformable models

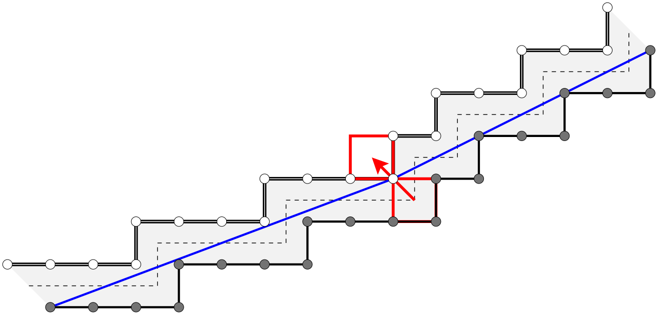


- G. Damiand, A. Dupas and J.-O. Lachaud, *Combining Topological Maps, Multi-Label Simple Points, and Minimum-Length Polygons for Efficient Digital Partition Model*, in proc. IWCIA2011, LNCS 6636, p. 208-221, 2011.

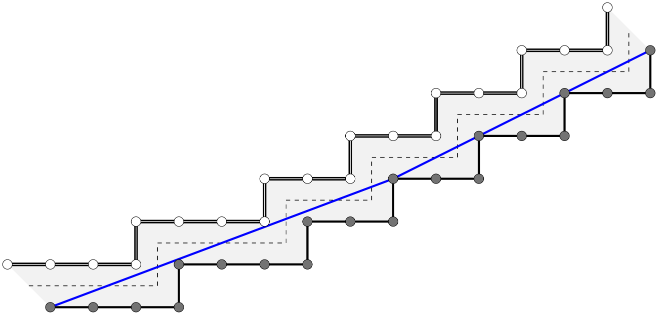
Flip a pixel



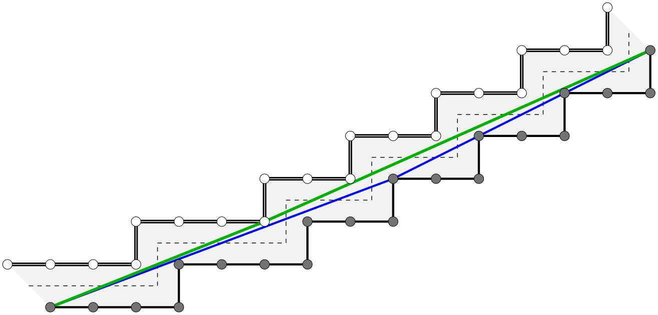
Flip a pixel



Flip a pixel



Flip a pixel



Reversible polygonal representation

Goal : represent a digital contour C using a polygon whose versices are centers of pixels either on the inner contour $\mathrm{IC}(C)$ or on the outer contour $\mathrm{OC}(C)$.

Reversible polygonal representation

Goal : represent a digital contour C using a polygon whose versices are centers of pixels either on the inner contour $\mathrm{IC}(C)$ or on the outer contour $\mathrm{OC}(C)$.

Definition

A *grid-vector* is a triplet $x = ((p, q), k, \delta) \in \mathbb{N}^2 \times \mathbb{N} \times \mathbb{B}$. where

- $\gcd(p, q) = 1$, q/p is the *slope* of x (with $1/0 = \infty$),
- $k \geq 1$ is its number of repetitions
- the boolean δ indicates if x has one endpoint on the inner contour and one on the outer.

Notation : $((p, q), k, \delta) = \begin{cases} (p, q)^k \text{ if } \delta \text{ is } \texttt{false}, \\ \\ \widetilde{(p, q)}^k \text{ otherwise.} \end{cases}$

Reversible polygonal representation

Geometric interpretation of grid-vectors.

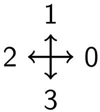
Definition

A *context* is an ordered pair of letters (a, b) among $\{(0,1),(1,2),(2,3),(3,0),(0,3),(3,2),(2,1),(1,0)\}$.

Given a context (a, b), a grid-vectors defines the following vector as follow :

$$\overset{(a,b)}{\overrightarrow{(p,q)^k}} = k(p\vec{a} + q\vec{b}),$$

$$\overset{(a,b)}{\overrightarrow{\widetilde{(p,q)}^k}} = k(p\vec{b} + q\vec{a}).$$



Reversible polygonal representation

Geometric interpretation of grid-vectors.

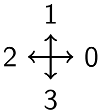
Definition

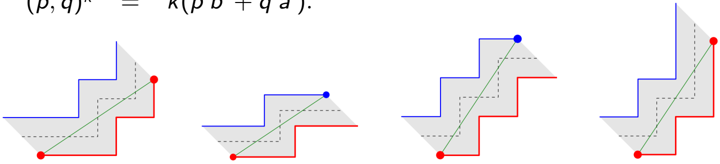
A *context* is an ordered pair of letters (a, b) among $\{(0,1),(1,2),(2,3),(3,0),(0,3),(3,2),(2,1),(1,0)\}$.

Given a context (a, b), a grid-vectors defines the following vector as follow :

$$\overrightarrow{(p,q)^k}^{(a,b)} = k(p\,\vec{a} + q\,\vec{b}),$$

$$\overrightarrow{\widetilde{(p,q)}^k}^{(a,b)} = k(p\,\vec{b} + q\,\vec{a}).$$



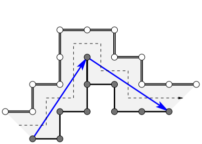


$(3,2)^1$ $\quad$ $\widetilde{(2,3)}^1$ $\quad$ $\widetilde{(3,2)}^1$ $\quad$ $(3,2)^1$

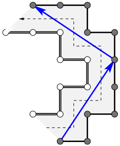
Reversible polygonal representation

- *Operators* : $\sigma^{+}(a,b) = (\overline{b}, a)$: a turn toward the interior,
  $\sigma^{-}(a,b) = (b, \overline{a})$: a turn toward the exterior,
  with the convention $\overline{0} = 2, \overline{1} = 3, \overline{2} = 0, \overline{3} = 1$.

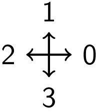
- *Grid-curve* : $\Gamma = [l_0, l_1, \ldots, l_{n-1}]$ where each l_i is either a grid-vector or one of the operators σ^{-}, σ^{+}.



$$[\underbrace{(2,3)}_{(0,1)}, \underbrace{\sigma^{+}}_{\substack{(0,1)\\ \downarrow \\ (3,0)}}, \underbrace{(2,3)}_{(3,0)}]$$



$$[\underbrace{(2,3)}_{(0,1)}, \underbrace{\sigma^{-}}_{\substack{(0,1)\\ \downarrow \\ (1,2)}}, \underbrace{(2,3)}_{(1,2)}]$$



Reversible polygonal representation

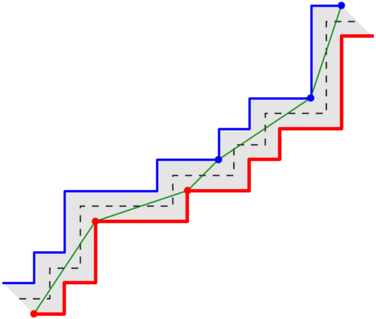
Notations :

- $\overset{(a,b)}{\overrightarrow{\sigma^-}} = \overset{(a,b)}{\overrightarrow{\sigma^+}} = (0,0)$.
- Let $x = ((p,q),k,\delta)$, $\quad x(a,b) = \begin{cases} (b,a) \text{ if } \delta \text{ is } \texttt{true}, \\ (a,b) \text{ otherwise.} \end{cases}$

Reversible polygonal representation

Notations :

- $\overset{(a,b)}{\overrightarrow{\sigma^-}} = \overset{(a,b)}{\overrightarrow{\sigma^+}} = (0,0)$.
- Let $x = ((p,q),k,\delta)$, $\quad x(a,b) = \begin{cases} (b,a) \text{ if } \delta \text{ is } \texttt{true}, \\ (a,b) \text{ otherwise.} \end{cases}$

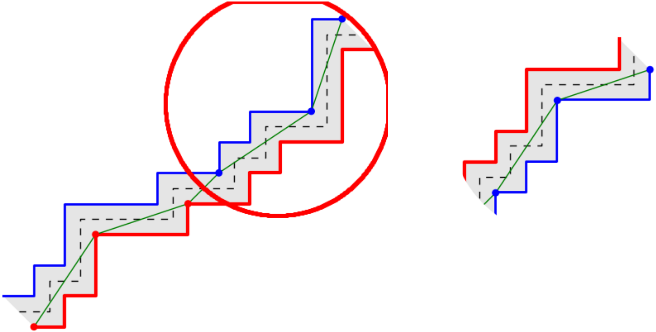


$[(2,3),(3,1),\widetilde{(1,1)},(2,3),(3,1)]$

Reversible polygonal representation

Notations :

- $\overset{(a,b)}{\overrightarrow{\sigma^-}} = \overset{(a,b)}{\overrightarrow{\sigma^+}} = (0,0)$.
- Let $x = ((p,q),k,\delta)$, $\quad x(a,b) = \begin{cases} (b,a) \text{ if } \delta \text{ is } \texttt{true}, \\ (a,b) \text{ otherwise.} \end{cases}$



$[(2,3),(3,1),\widetilde{(1,1)},(2,3),(3,1)]$

Notations :

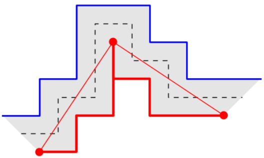
- $\overset{(a,b)}{\overrightarrow{\sigma^-}} = \overset{(a,b)}{\overrightarrow{\sigma^+}} = (0,0)$.
- Let $x = ((p,q),k,\delta)$, $\quad x(a,b) = \begin{cases} (b,a) \text{ if } \delta \text{ is } \texttt{true}, \\ (a,b) \text{ otherwise.} \end{cases}$

From grid-curves to polygons.
A grid-curve $\Gamma = [l_0, l_1, \ldots, l_{n-1}]$, a context (a_0, b_0) and a start point P_0 define a polygonal curve $P_\Gamma = [P_0, P_1, \ldots, P_n]$ in the following way :

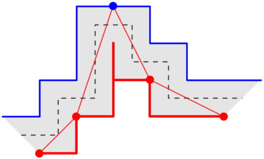
$$P_{i+1} = P_i + \overset{(a_i,b_i)}{\overrightarrow{l_i}} \quad \text{and} \quad (a_{i+1}, b_{i+1}) = l_i(a_i, b_i).$$

By fixing the first point on the inside or outside polygon, a discrete contour is defined unambiguously.

Not unique

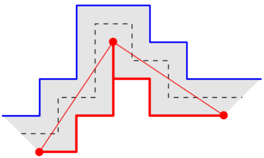


$[(2,3), \sigma^+, (2,3)]$

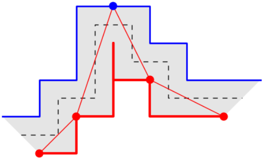


$[(1,1), \widetilde{(3,1)}, \sigma^-, \widetilde{(2,1)}, (1,2)]$

Not unique



$[(2,3), \sigma^+, (2,3)]$



$[(1,1), \widetilde{(3,1)}, \sigma^-, \widetilde{(2,1)}, (1,2)]$

Definition

Two grid-curves Γ and Γ' are *equivalent*, if they define the same digital contour and ends in the same orientation.

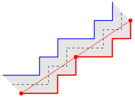
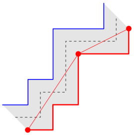
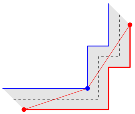
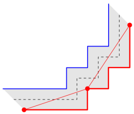
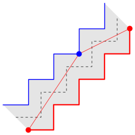
The MLP of the digital contour C is the shortest grid-curve in the equivalence class defined by C.

Relative orientation of grid-segements

Notation

Given $x = ((p,q), k, \delta_x)$ and $y = ((r,s), l, \delta_y)$,

$$x \otimes y = \begin{cases} ps - qr & \text{if } \delta_y \text{ is } \texttt{false}, \\ pr - qs & \text{if } \delta_y \text{ is } \texttt{true}. \end{cases}$$

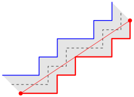
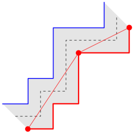
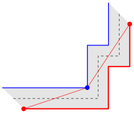
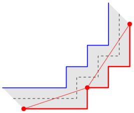
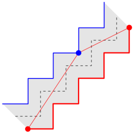
Three cases				
$x \otimes y = 0$	$x \otimes y < 0$		$x \otimes y > 0$	
				
$[(3,2),(3,2)]$	$[(2,3),(2,1)]$	$[\widetilde{(1,3)},\widetilde{(2,3)}]$	$[(3,1),(2,3)]$	$[\widetilde{(3,2)},\widetilde{(2,1)}]$

Relative orientation of grid-segements

Notation

Given $x = ((p, q), k, \delta_x)$ and $y = ((r, s), l, \delta_y)$,

$$x \otimes y = \begin{cases} ps - qr & \text{if } \delta_y \text{ is } \texttt{false}, \\ pr - qs & \text{if } \delta_y \text{ is } \texttt{true}. \end{cases}$$

Three cases				
$x \otimes y = 0$	$x \otimes y < 0$		$x \otimes y > 0$	
				
$[(3, 2)^2]$	$[(2, 3), (2, 1)]$	$[\widetilde{(1, 3)}, \widetilde{(2, 3)}]$	$[(3, 1), (2, 3)]$	$[\widetilde{(3, 2)}, \widetilde{(2, 1)}]$

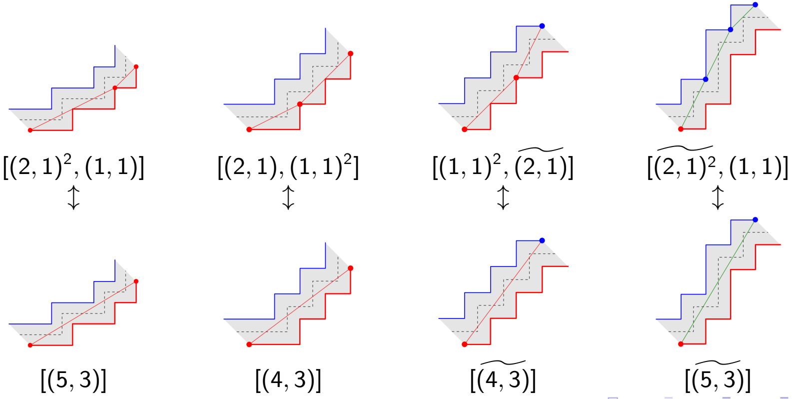
Merge case : $x \otimes y = 1$

Let $x = ((p, q), k, \delta_x)$ and $y = ((r, s), l, \delta_y)$ with

$$x \otimes y = 1, \qquad \begin{array}{c} \delta_y = \texttt{false} \text{ and } \min(k, l) = 1 \\ \text{or} \\ \delta_y = \texttt{true} \text{ and } l = 1 \end{array}$$

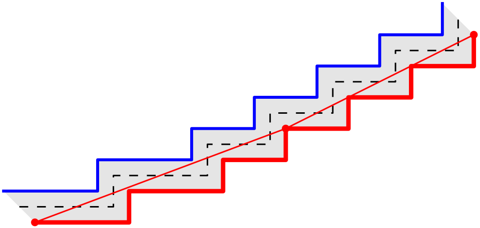
then

$$[x, y] \equiv [z] \text{ where } z = \begin{cases} ((kp + lr, kq + ls), 1, \delta_x) & \text{if } \delta_y = \texttt{false}. \\ ((kp + ls, kq + lr), 1, \neg\delta_x) & \text{otherwise}. \end{cases}$$



Split and merge case : $x \otimes y > 1$

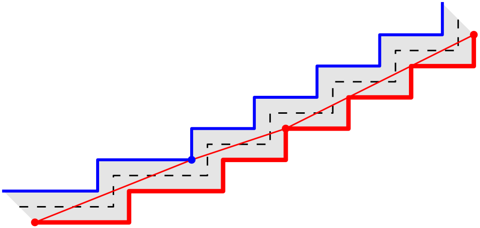
$(8,3) \otimes (2,1)^3 = 2.$



$[(8,3),(2,1)^3]$

Split and merge case : $x \otimes y > 1$

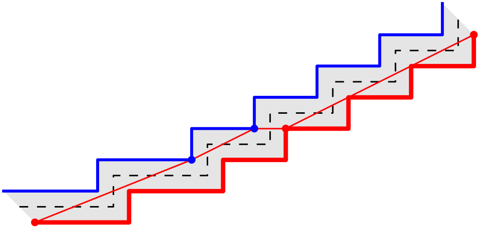
$(8,3) \otimes (2,1)^3 = 2.$



$[(8,3),(2,1)^3] \equiv [\widetilde{(2,5)},\widetilde{(3,1)},(2,1)^3]$

Split and merge case : $x \otimes y > 1$

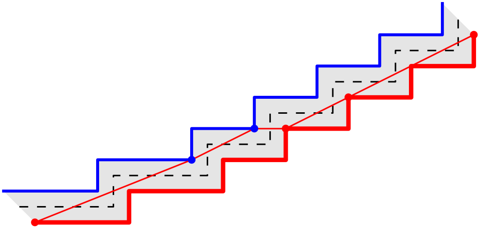
$(8,3) \otimes (2,1)^3 = 2.$



$[(8,3),(2,1)^3] \equiv [\widetilde{(2,5)},(1,2),\widetilde{(1,0)},(2,1)^3]$

Split and merge case : $x \otimes y > 1$

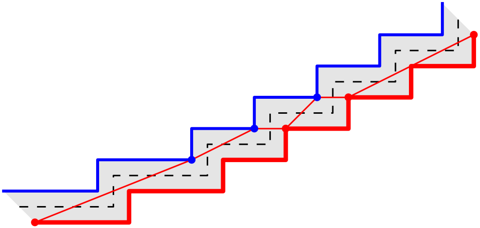
$(8,3) \otimes (2,1)^3 = 2.$



$[(8,3),(2,1)^3] \equiv [\widetilde{(2,5)},(1,2),\widetilde{(1,0)},(2,1),(2,1)^2]$

Split and merge case : $x \otimes y > 1$

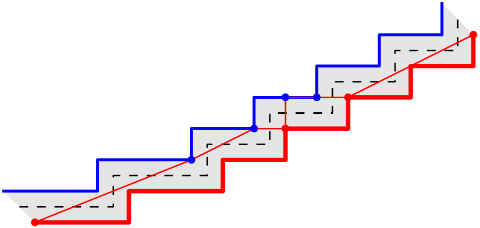
$$(8,3) \otimes (2,1)^3 = 2.$$



$$[(8,3),(2,1)^3] \equiv [\widetilde{(2,5)},(1,2),\widetilde{(1,0)},\widetilde{(1,1)},\widetilde{(1,0)},(2,1)^2]$$

Split and merge case : $x \otimes y > 1$

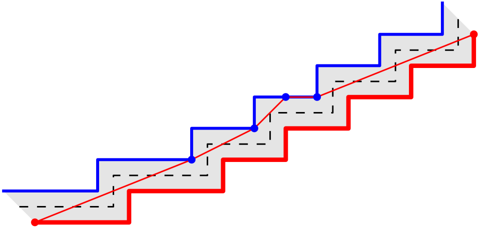
$$(8,3) \otimes (2,1)^3 = 2.$$



$$[(8,3),(2,1)^3] \quad \equiv [\widetilde{(2,5)},(1,2),\widetilde{(1,0)},\widetilde{(1,0)},(0,1),\widetilde{(1,0)},(2,1)^2]$$

Split and merge case : $x \otimes y > 1$

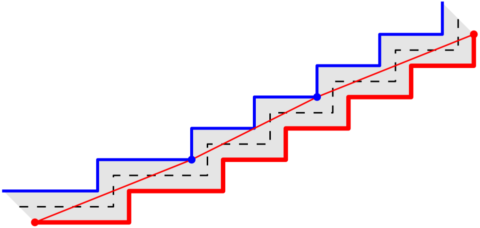
$(8,3) \otimes (2,1)^3 = 2.$



$[(8,3),(2,1)^3] \quad \equiv [\widetilde{(2,5)},(1,2),(1,1)(0,1),\widetilde{(5,2)}]$

Split and merge case : $x \otimes y > 1$

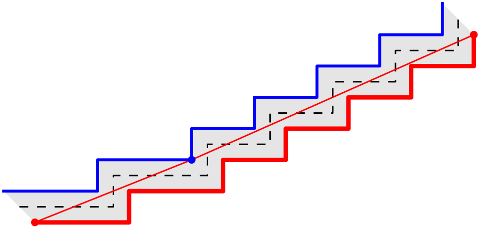
$$(8,3) \otimes (2,1)^3 = 2.$$



$$[(8,3),(2,1)^3] \quad \equiv [\widetilde{(2,5)},(1,2)^2,\widetilde{(5,2)}]$$

Split and merge case : $x \otimes y > 1$

$$(8,3) \otimes (2,1)^3 = 2.$$



$$[(8,3),(2,1)^3] \equiv [\widetilde{(2,5)}, \widetilde{(9,4)}]$$

How to split ?

Notation

Let $x = ((p, q), 1, \texttt{false})$ and $q/p = [u_0; u_1, \ldots, u_n]$.

- $q_i/p_i = [u_0; u_1, \ldots, u_i]$,
- $x_i = ((p_i, q_i), 1, \texttt{false})$,
- $x_{-1} = ((0, 1), 1, \texttt{false})$,
- $x_{-2} = ((1, 0), 1, \texttt{false})$.

Definition

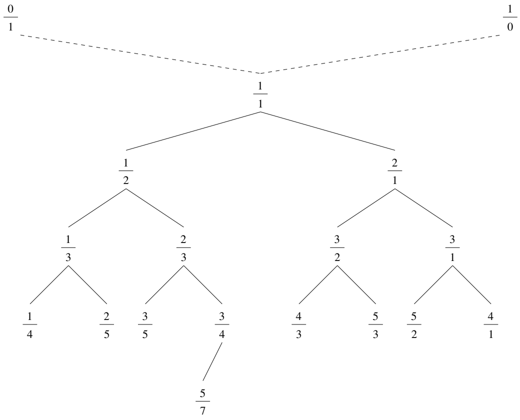
The *basic splitting* of the grid-vector x_n is the grid-curve :

$$s(x_n) = \begin{cases} [x_{2m-2}, x_{2m-1}^{u_{2m}}] \text{ if n = 2m}, \\ [x_{2m}^{u_{2m+1}}, x_{2m-1}] \text{ if n = 2m+1}, \end{cases}$$

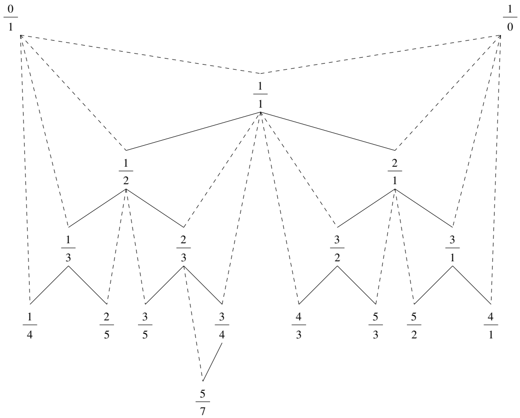
A grid-vector and it's basic splittings both define the same interpixel path.

$$s(x) = [y, z] \implies y \otimes z = 1.$$

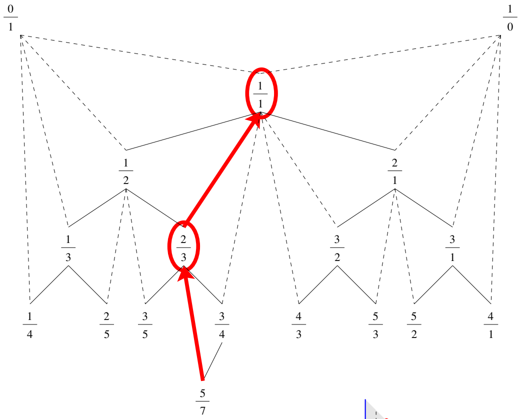
How to split ?



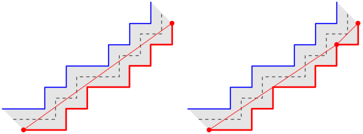
How to split ?



How to split ?

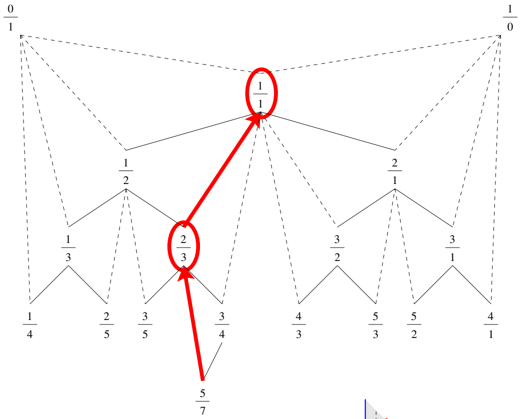


$5/7 = [0; 1, 2, 2]$,
$2/3 = [0; 1, 2]$,
$1/1 = [0; 1]$



$$[(7, 5)] \equiv [(3, 2)^2, (1, 1)]$$

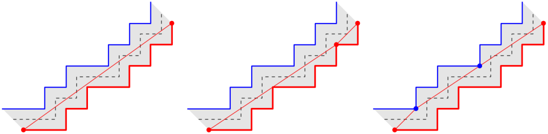
How to split ?



$5/7 = [0; 1, 2, 2],$

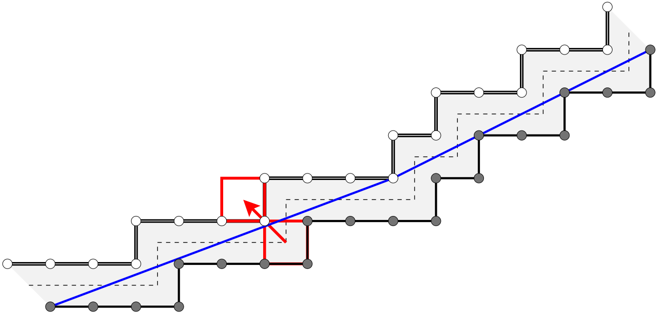
$2/3 = [0; 1, 2],$

$1/1 = [0; 1]$



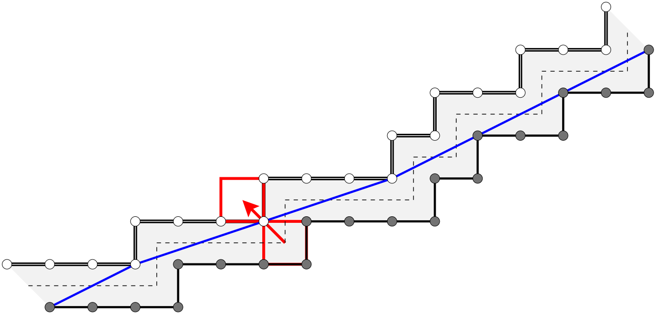
$$[(7,5)] \equiv [(3,2)^2, (1,1)] \equiv [\widetilde{(1,1)}, (2,3), \widetilde{(3,2)}]$$

Flip a pixel



$[\widetilde{(3,8)}, \widetilde{(2,1)^3}]$

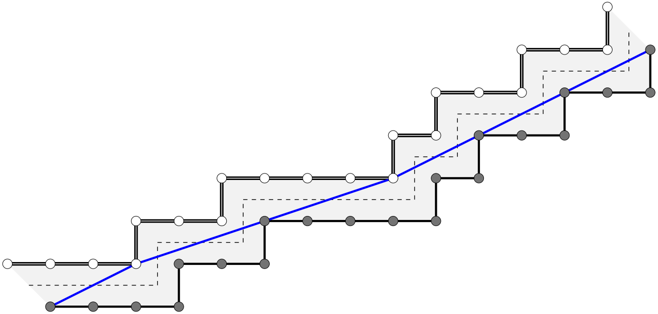
Flip a pixel



$$[\widetilde{(3,8)}, \widetilde{(2,1)^3}] \equiv [\widetilde{(1,2)}, (1,3), (1,3), \widetilde{(2,1)^3}]$$

1. Split grid-segments until one ends exactly on the pixel to flip. Let $x = ((p,q), 1, \delta_x)$ be the grid segment right before and $y = ((r,s), 1, \delta_y)$ be the grid-vector right after.

Flip a pixel

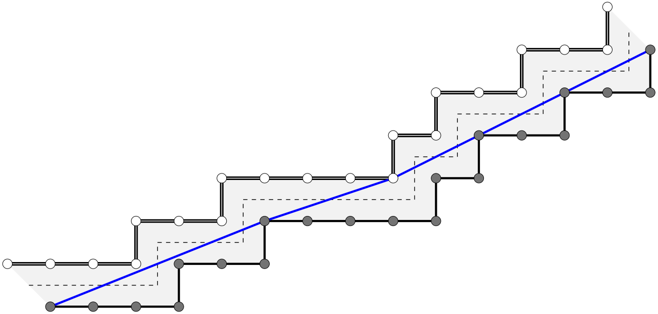


$$[\widetilde{(3,8)}, \widetilde{(2,1)}^3] \equiv [\widetilde{(1,2)}, (1,3), (1,3), \widetilde{(2,1)}^3]$$

$$\not\equiv [\widetilde{(1,2)}, \widetilde{(3,1)}, \widetilde{(1,3)}, \widetilde{(2,1)}^3]$$

1. Split grid-segments until one ends exactly on the pixel to flip. Let $x = ((p,q), 1, \delta_x)$ be the grid segment right before and $y = ((r,s), 1, \delta_y)$ be the grid-vector right after.
2. Replace x by $((q,p), 1, \neg\delta_x)$.
3. Replace y by $((r,s), 1, \neg\delta_y)$.

Flip a pixel

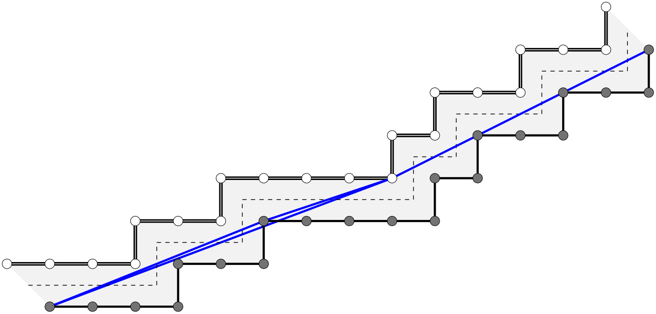


$$[\widetilde{(3,8)}, \widetilde{(2,1)^3}] \equiv [\widetilde{(1,2)}, (1,3), (1,3), \widetilde{(2,1)^3}]$$

$$\not\equiv [\widetilde{(1,2)}, \widetilde{(3,1)}, \widetilde{(1,3)}, \widetilde{(2,1)^3}] \equiv [(5,2), \widetilde{(1,3)}, \widetilde{(2,1)^3}]$$

1. Split grid-segments until one ends exactly on the pixel to flip. Let $x = ((p,q), 1, \delta_x)$ be the grid segment right before and $y = ((r,s), 1, \delta_y)$ be the grid-vector right after.
2. Replace x by $((q,p), 1, \neg\delta_x)$.
3. Replace y by $((r,s), 1, \neg\delta_y)$.

Flip a pixel

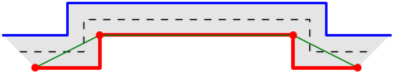


$$[\widetilde{(3,8)}, \widetilde{(2,1)^3}] \equiv [\widetilde{(1,2)}, (1,3), (1,3), \widetilde{(2,1)^3}]$$

$$\not\equiv [\widetilde{(1,2)}, \widetilde{(3,1)}, \widetilde{(1,3)}, \widetilde{(2,1)^3}] \equiv [(5,2), \widetilde{(1,3)}, \widetilde{(2,1)^3}]$$

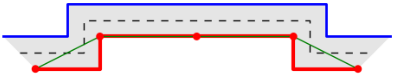
1. Split grid-segments until one ends exactly on the pixel to flip. Let $x = ((p,q), 1, \delta_x)$ be the grid segment right before and $y = ((r,s), 1, \delta_y)$ be the grid-vector right after.
2. Replace x by $((q,p), 1, \neg\delta_x)$.
3. Replace y by $((r,s), 1, \neg\delta_y)$.

Flip a pixel on a flat part



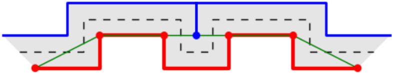
$$[(2,1),(1,0)^6,\sigma^+,(1,2)]$$

Flip a pixel on a flat part



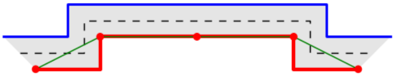
$$[(2,1),(1,0)^3,(1,0)^3,\sigma^+,(1,2)]$$

Flip a pixel on a flat part



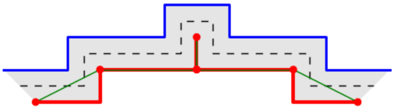
$$[(2,1),(1,0)^2,\widetilde{(0,1)},\widetilde{(1,0)^3},\sigma^+,(1,2)]$$

Flip a pixel on a flat part



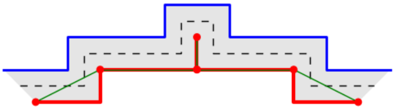
$$[(2,1),(1,0)^3,(1,0)^3,\sigma^+,(1,2)]$$

Flip a pixel on a flat part



$$[(2,1),(1,0)^3, \quad \underbrace{\sigma^-,(1,0),\sigma^+,\sigma^+,(1,0),\sigma^-,}_{\text{bumb}} \quad (1,0)^3,\sigma^+,(1,2)]$$

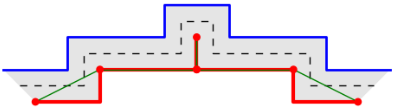
Flip a pixel on a flat part



$$[(2,1),(1,0)^3, \quad \underbrace{\sigma^-,(1,0),\sigma^+,\sigma^+,(1,0),\sigma^-,}_{\text{bumb}} \quad (1,0)^3,\sigma^+,(1,2)]$$

How to simplify σ^- ?

Flip a pixel on a flat part

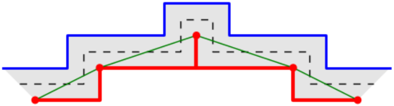


$$[(2,1),(1,0)^3, \quad \underbrace{\sigma^-,(1,0),\sigma^+,\sigma^+,(1,0),\sigma^-,}_{\text{bumb}} \quad (1,0)^3,\sigma^+,(1,2)]$$

How to simplify σ^- ?

- Cancellation : $[\sigma^-,\sigma^+] \equiv [\sigma^+,\sigma^-] \equiv [\,]$

Flip a pixel on a flat part



$$[(2,1),(1,0)^3,\quad \underbrace{\sigma^-,(1,0),\sigma^+,\sigma^+,(1,0),\sigma^-,}_{\text{bumb}}\quad (1,0)^3,\sigma^+,(1,2)]$$

How to simplify σ^- ?

- Cancellation : $[\sigma^-,\sigma^+] \equiv [\sigma^+,\sigma^-] \equiv [\,]$
- Split the grid-edges in order to have a local part build only with $\{\sigma^+,\sigma^-,(1,0),(0,1),\widetilde{(1,0)},\widetilde{(0,1)}\}$. Operators σ^- are then simplify using local rules such as :

$$[(1,0),\sigma^-,(1,0),\sigma^+] \equiv [(1,1)] \text{ and } [\sigma^-,(1,0)^k,\sigma^+] \equiv [(0,1)^k]$$

Main result

Proposition

A grid-curve defining a digital contour may be simplified to a MLP using local rules.

Main result

Proposition

A grid-curve defining a digital contour may be simplified to a MLP using local rules.

Proposition

Given a grid-curve that is the MLP of a digital contour, this contour may be modified by adding or removing one pixel and its MLP updated in time sub-linear with respect to the length of the modified part of the MLP.

Implemente in project *ImaGene* available at

`gforge.liris.cnrs.fr/projects/imagene`

MERCI !