Chapitre 5 : Segmentation (Partie 1)

Joël Lefebvre (UQÀM)

INF600F – Traitement d’images

Automne 2024

Annonces

- TP3 en ligne bientôt
- Laboratoire vendredi (Segmentation)

Survol du cours

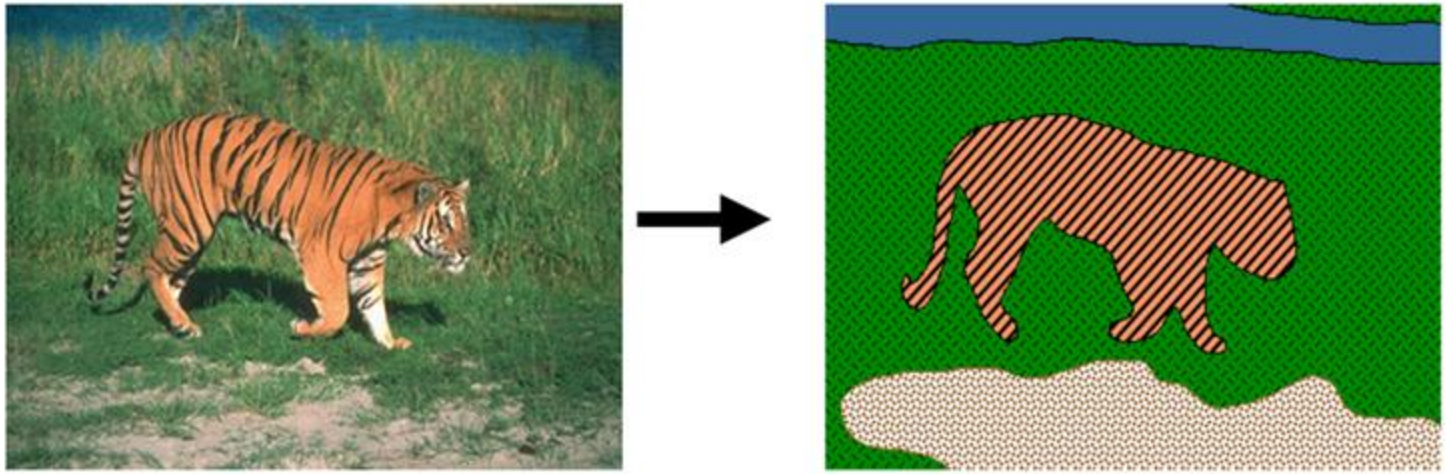
- Introduction
- Détection de contours
- Détection de courbes simples
- Segmentation de régions (prochain cours)

Références

- (Chityala, 2020) Section 4.3 : *Edge Detection using Derivatives*
- (Chityala, 2020) Section 10.3 : *Hough Transform*
- (Burger, 2009) Vol1 Ch6 : *Edges and Contours*
- (Burger, 2009) Vol2 Ch2 : *Detecting Simple Curves*
- (Gonzalez, 2018) Chapitre 10

Qu’est-ce que la segmentation ?

- **Regroupement** de pixels en objets
- « **Organisation perceptuelle** »
- Chaque région doit satisfaire une **propriété commune**
- Deux régions voisines ne doivent pas satisfaire la même propriété



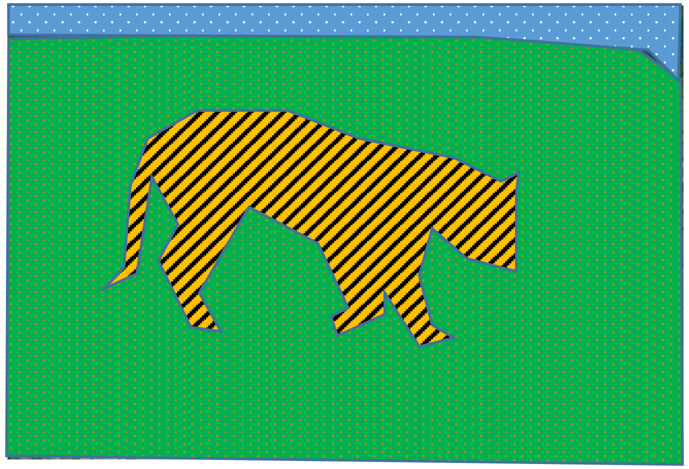
Source : N. Stikov & N. Snavely

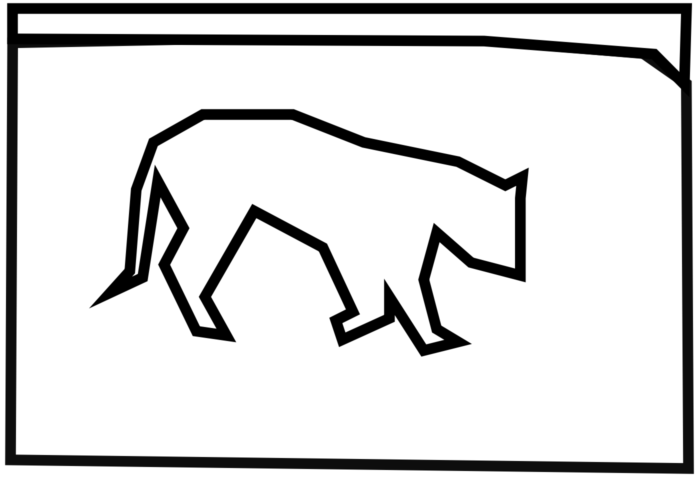
Pourquoi regrouper les pixels ?

- **Pixels** : Propriété des détecteurs, pas de la scène
- Travailler au niveau des **objets** peut rendre le traitement plus facile
  - Les objets sont à une **profondeur** constante
  - Les objets peuvent être **reconnus**
  - Les objets peuvent se **déplacer en bloc**

Source : N. Snavely

Rappel : Région / Frontière





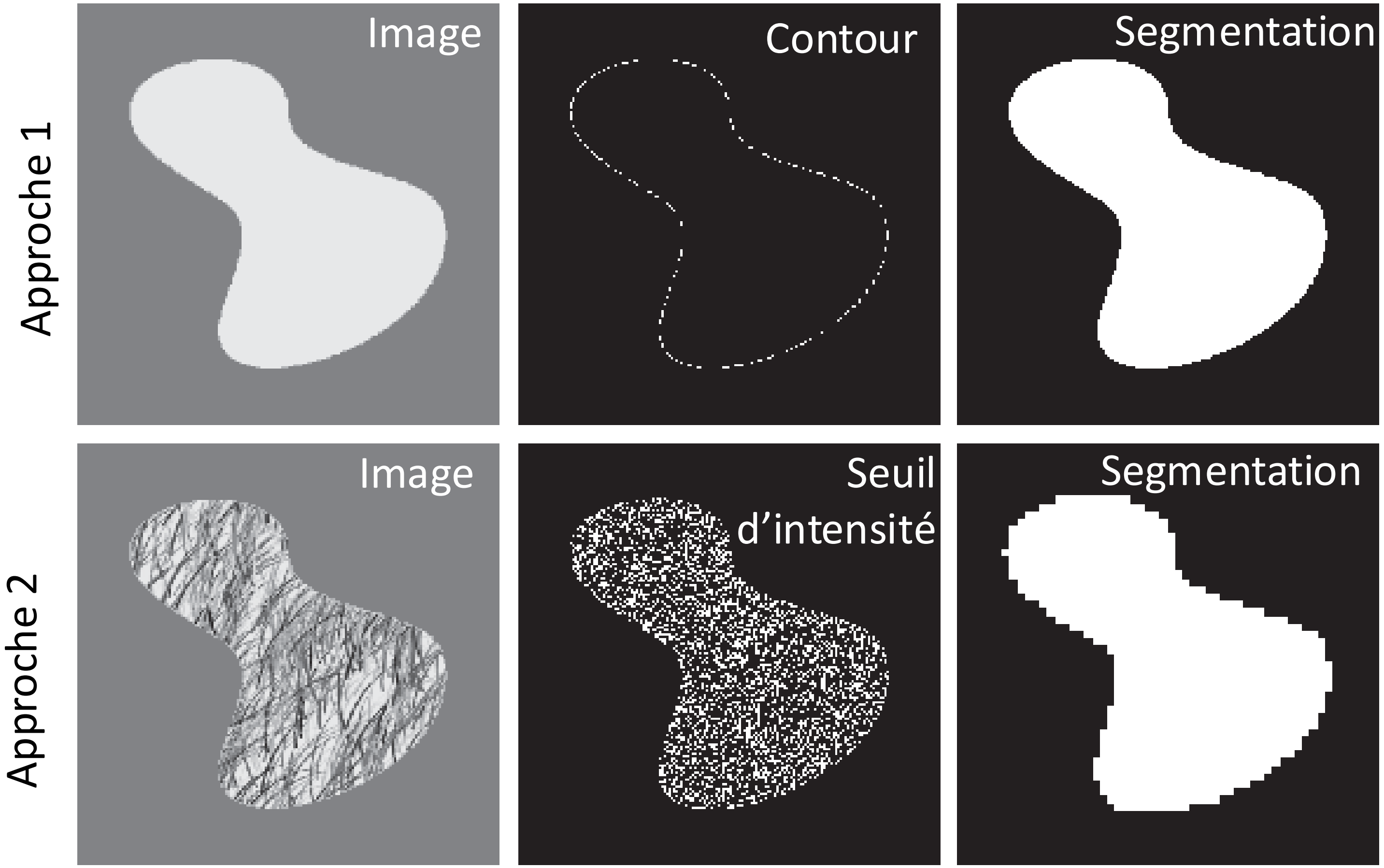
Source : N. Snavely

Dualité contour / région

- **Approche 1 :** Détecter un objet à l’aide de son **contour**
- **Approche 2 :** Détecter un objet à l’aide de l’apparence de sa **région**

<u>Formulations équivalentes ?</u>

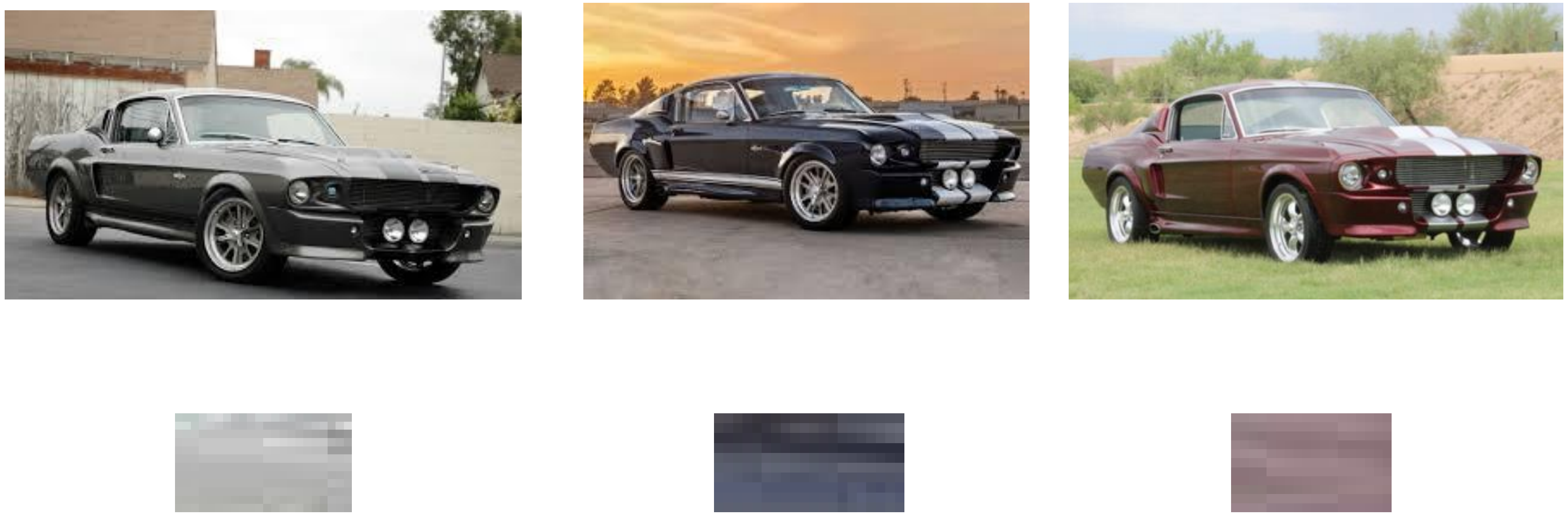
- En théorie : oui
- En pratique : non



(Gonzalez, Woods, 4e)

Pourquoi utiliser les contours?

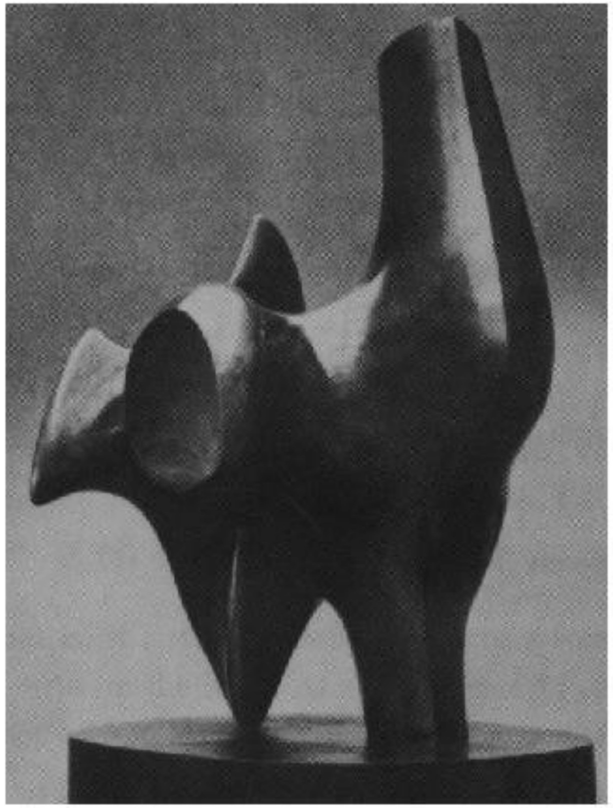
- Résilience aux **changements** de couleur / illumination
- Utile pour la **reconnaissance** d'images
- Utile pour la **correspondance** de sous-régions (***patch***)

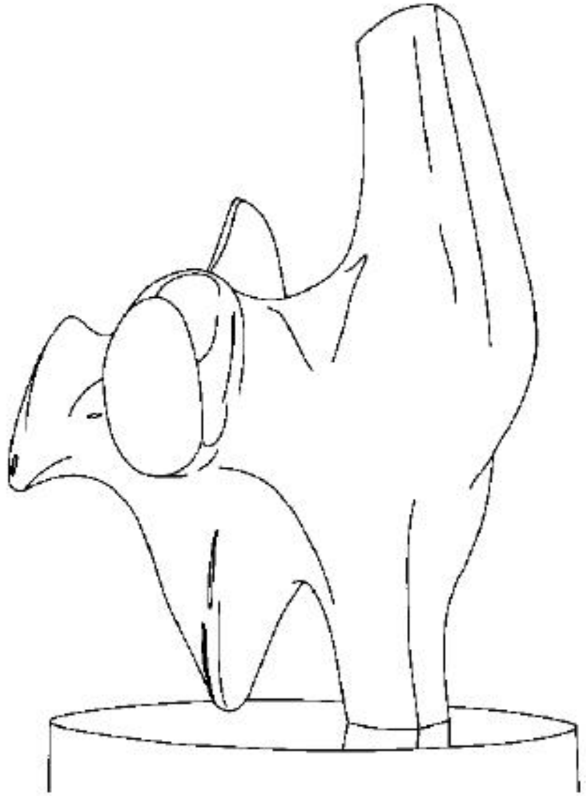


Source : N. Snavely

Argument psychophysique
Pourquoi les contours ?

- La **vision humaine** est très sensible aux contours
- Conversion d’une image 2D en un ensemble de courbes
- Extraction des **éléments saillants** de la scène, représentation **compacte**

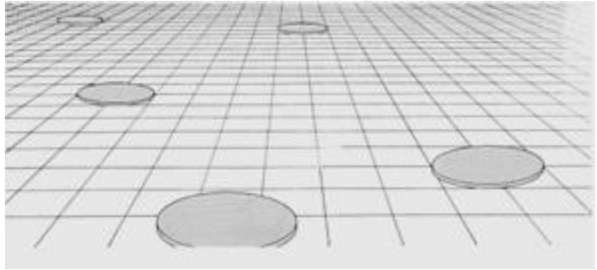




Source : N. Snavely

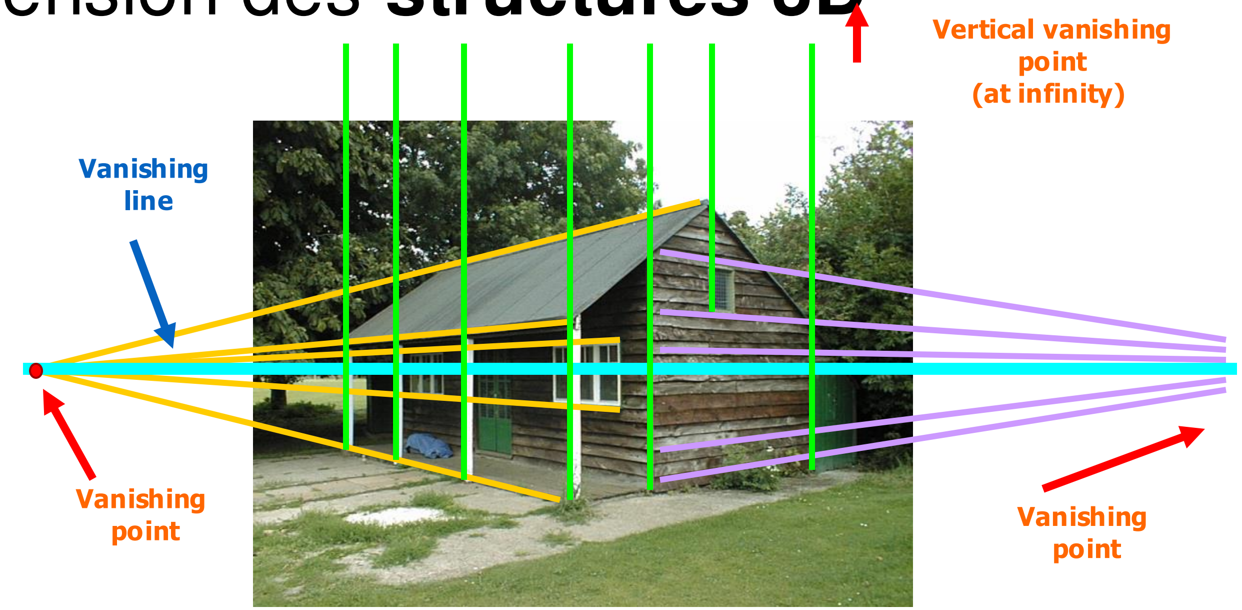
Argument de vision par ordinateur
Pourquoi les contours ?

- Fournit des **indices** sur la forme et la **géométrie**
- Utile pour la **reconnaissance** d'objets
- Utile pour la compréhension des **structures 3D**



Credit: Jitendra Malik

Credit: Attneave



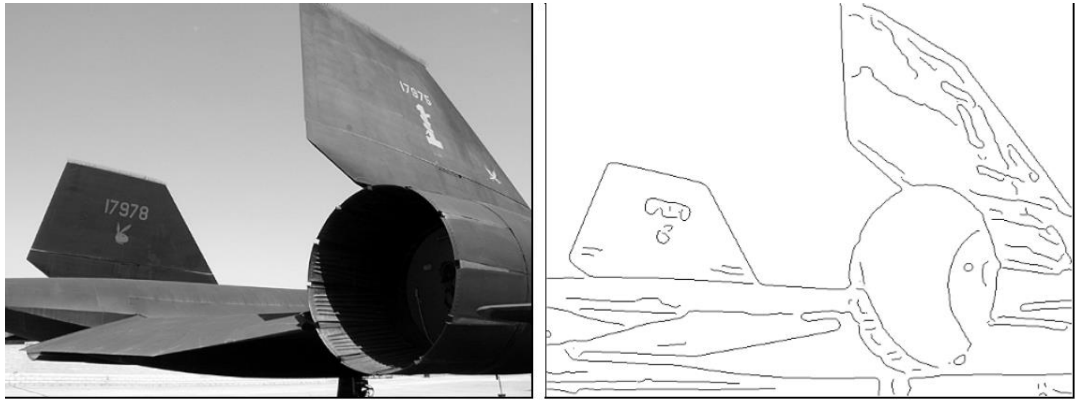
Source : N. Snavely

Détection de contours

Chapitre 5 : Segmentation (Partie 1)

Joël Lefebvre (UQÀM)

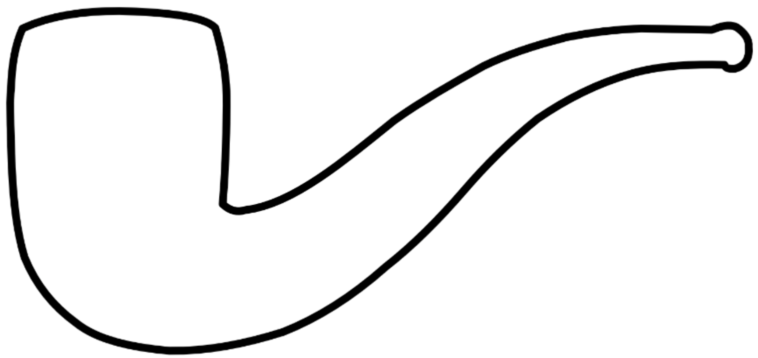
INF600F – Traitement d'images



(Burger 2009, Fig 6.1)

Détection de contours – Aperçu

- **But :** Identifier les changements visuels dans une image
- Intuitivement, de l’information **sémantique** est encodée dans les **contours**.
- **Quelles sont les « causes » des contours visuels?**



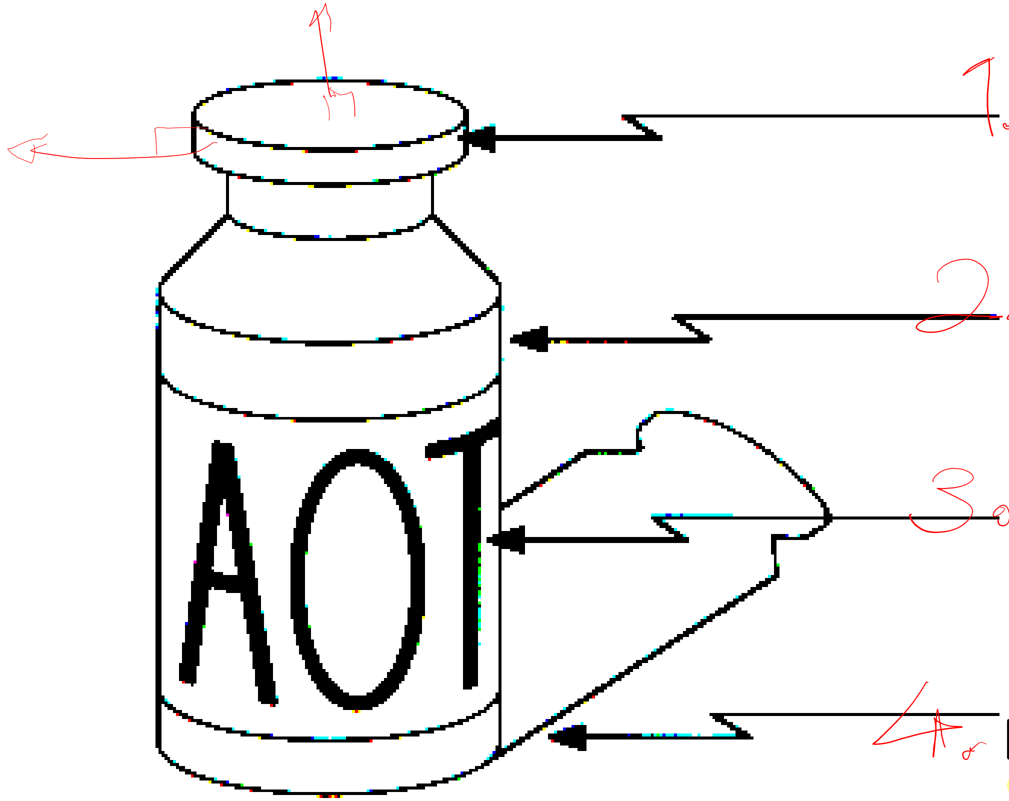
Ceci est une pipe

Adapté de la peinture « La trahison des images » de René Margritte (1929, Source)

Origine des contours

c'est un vector qui point dehors et 90°, indique la direction

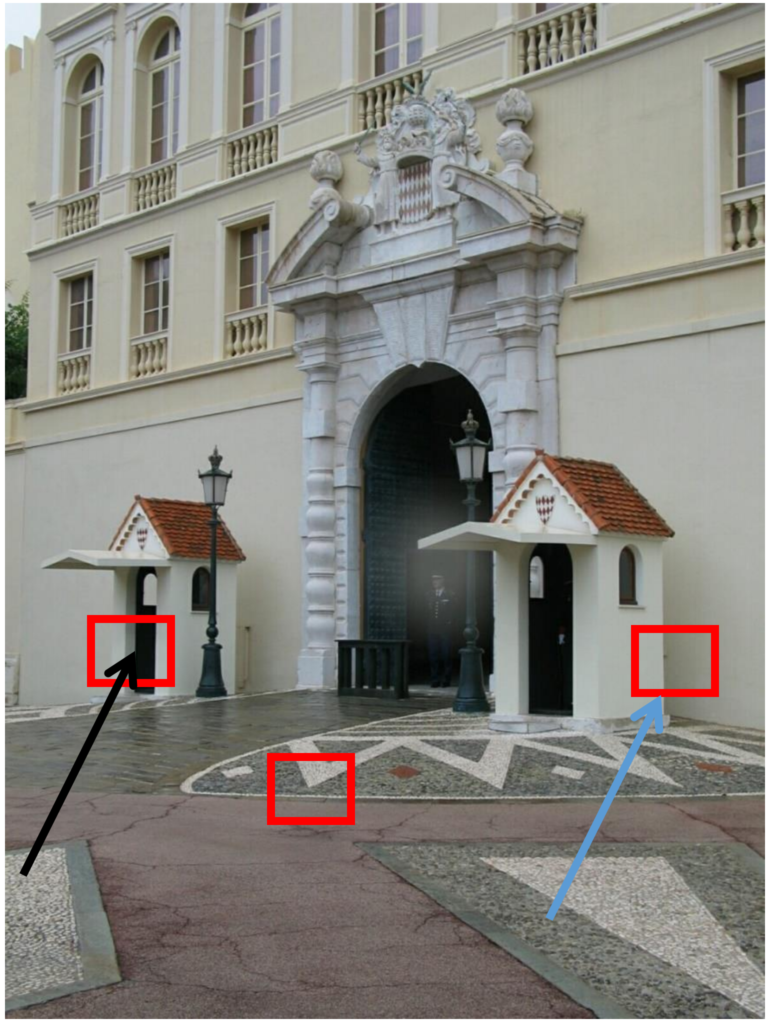
- Les contours sont causés par une variété de facteurs

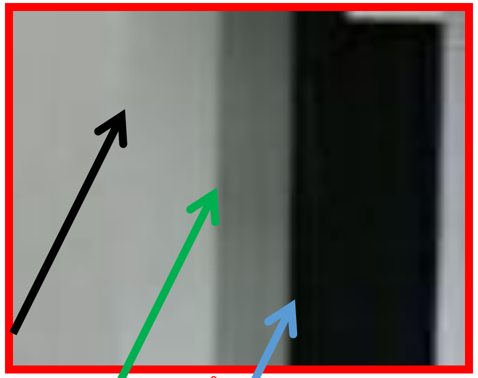


1. Discontinuité des **normales** de surface
2. Discontinuité de **profondeur**
3. Discontinuité de **couleur** de surface
4. Discontinuité **d'illumination**

Source: Steve Seitz

Exemple de contours





normal

prof

color

Source: D. Hoiem

Caractérisation des contours (1D)

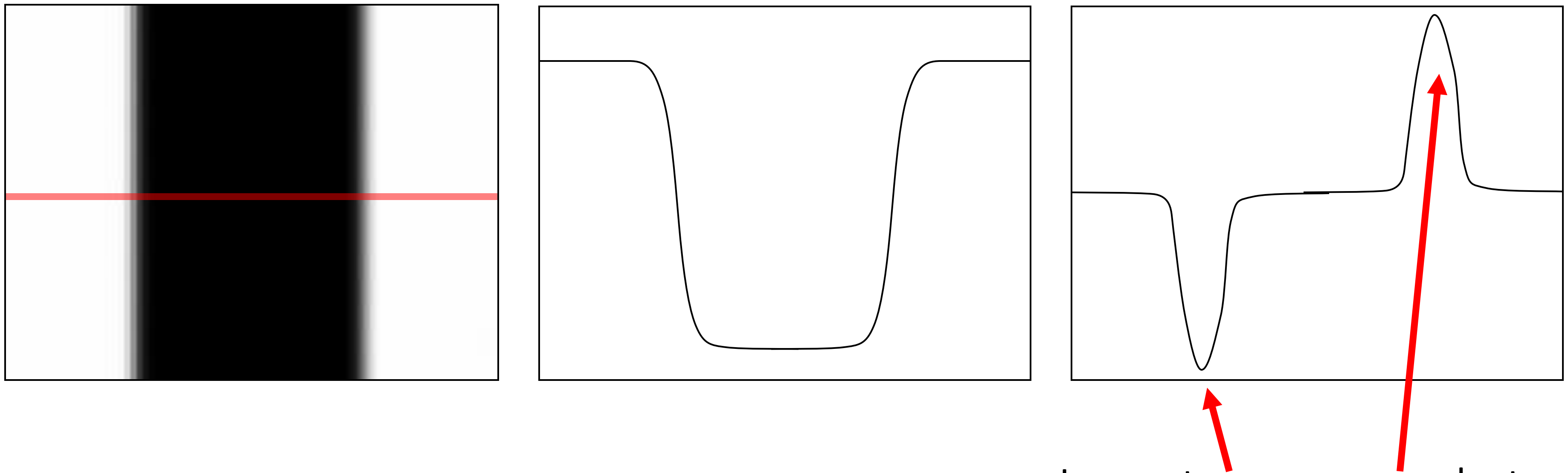
- Un contour est un endroit présentant un **changement rapide** de l'intensité de l'image

$$\frac{df}{dx} = \lim_{x \to 0} \frac{f(x+h)-f(x)}{h}$$

Image

Intensité
(le long d'une ligne horizontale)

1ère dérivée



Les contours correspondent aux **extremas** de la dérivée

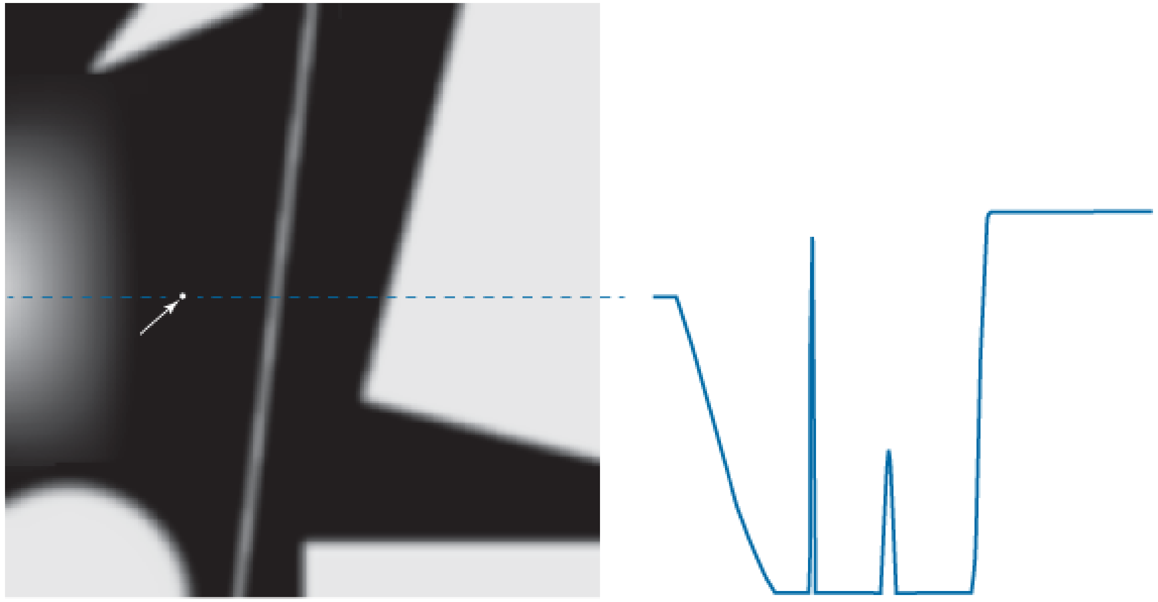
Source: L. Lazebnik & N. Snavely

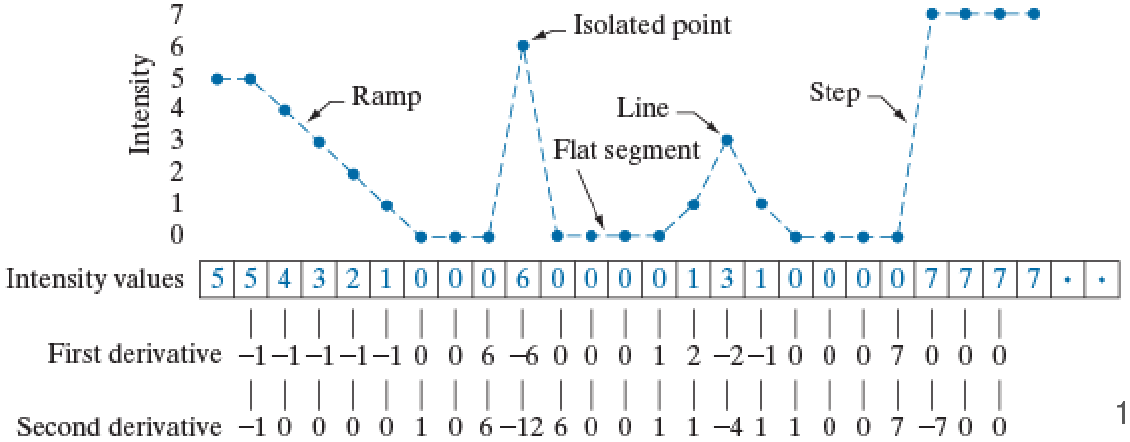
Exemples de contour 1D

<u>**Plusieurs types de segments**</u>

- Variation lente (*ramp*)
- Point isolé
- Région uniforme (*flat*)
- Ligne
- Échelle (*step*)

Chaque type de segments influence la valeur et les variations (dérivées) du profil





Méthode du gradient (1D)

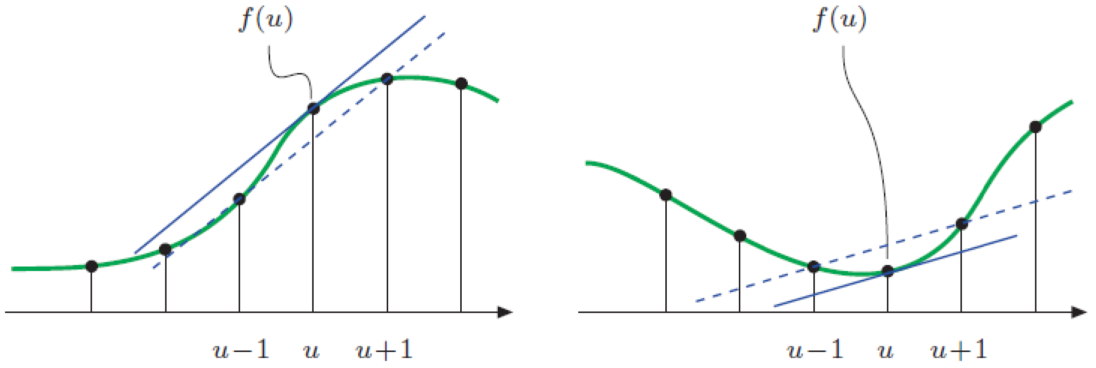
$\frac{1}{2}[-1\ 0\ 1]$

- Intensité le long d'une ligne : $f(x)$
- **1^ere^ dérivée** du signal : $f'(x) = \frac{df}{dx}(x)$
- On doit **approximer** la dérivée pour un **signal discret**

$f : [0\ 0\ 1\ 1\ 2\ 0\ \dots]$

$\frac{1}{2}\quad \frac{1}{2}\quad 0\quad -1$

$$\frac{df}{du}(u) \approx \frac{f(u+1) - f(u-1)}{(u+1) - (u-1)} = \frac{f(u+1) - f(u-1)}{2}$$

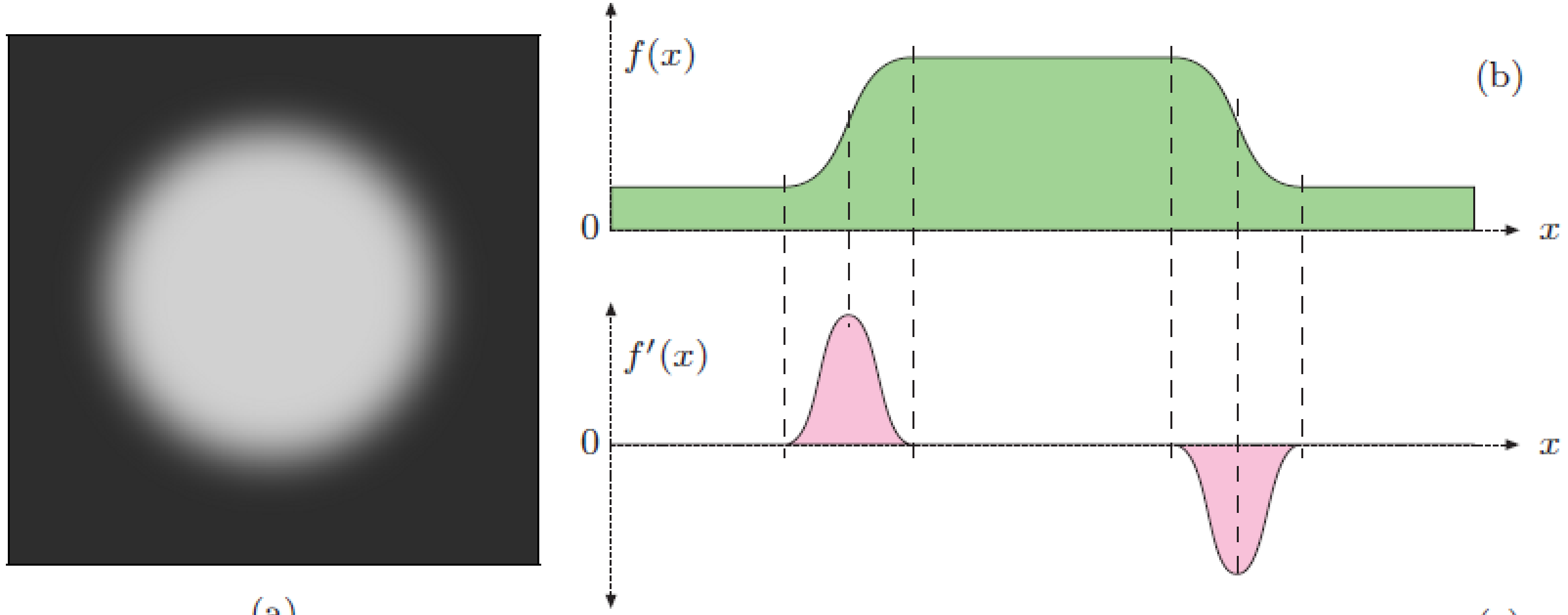


(Burger, 2009)

Méthode du **gradient** (2D)

<u>Dérivée partielle :</u> Dérivée selon une direction donnée pour une fonction multidimensionnelle $f(x, y)$

- $\frac{\partial f}{\partial x}$: Dérivée partielle selon la direction x
- $\frac{\partial f}{\partial y}$: Dérivée partielle selon la direction y



(a)
(c)

Gradient de l’image

- Le gradient est un **vecteur** qui pointe dans la direction de l’augmentation d’intensité la plus rapide

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

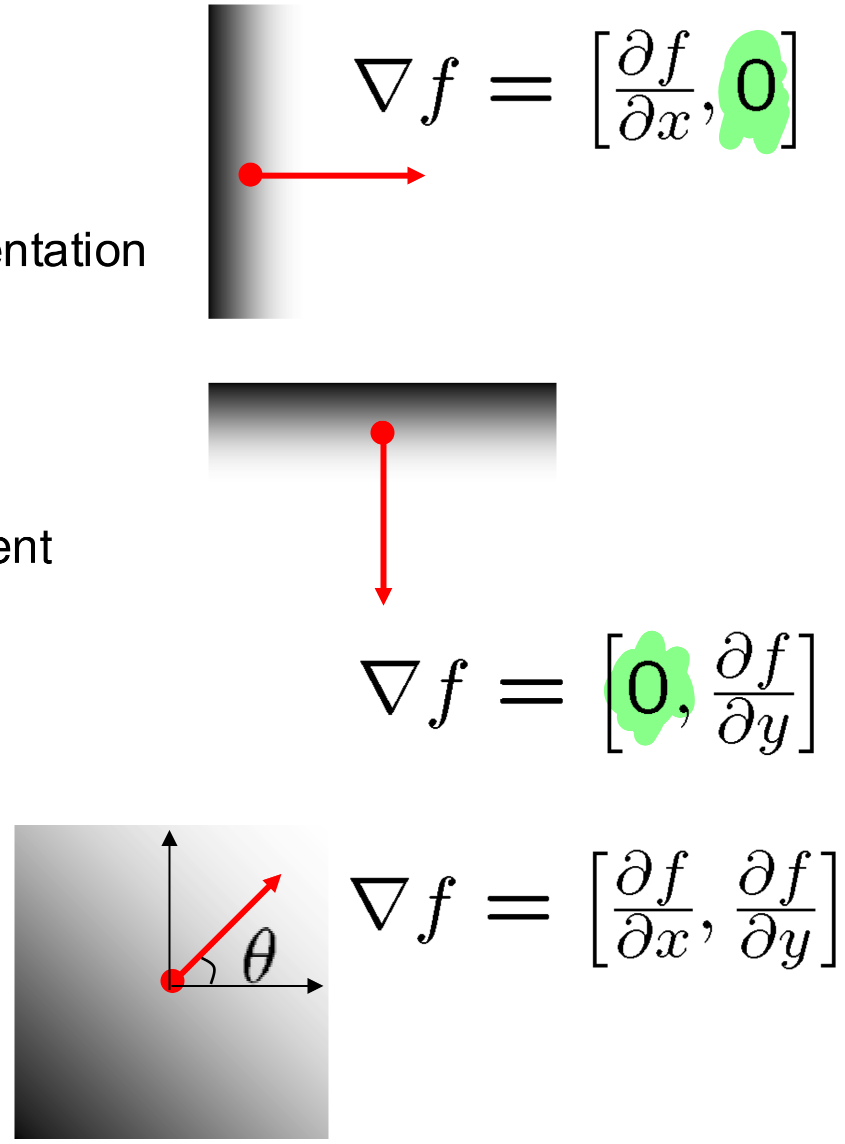
- La ***force*** du contour est donnée par **l’amplitude** du vecteur gradient

$$|\nabla f| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

- La ***direction*** du gradient est donnée par

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} \Big/ \frac{\partial f}{\partial x}\right)$$

- **Quel est le lien avec la direction des contours ?**



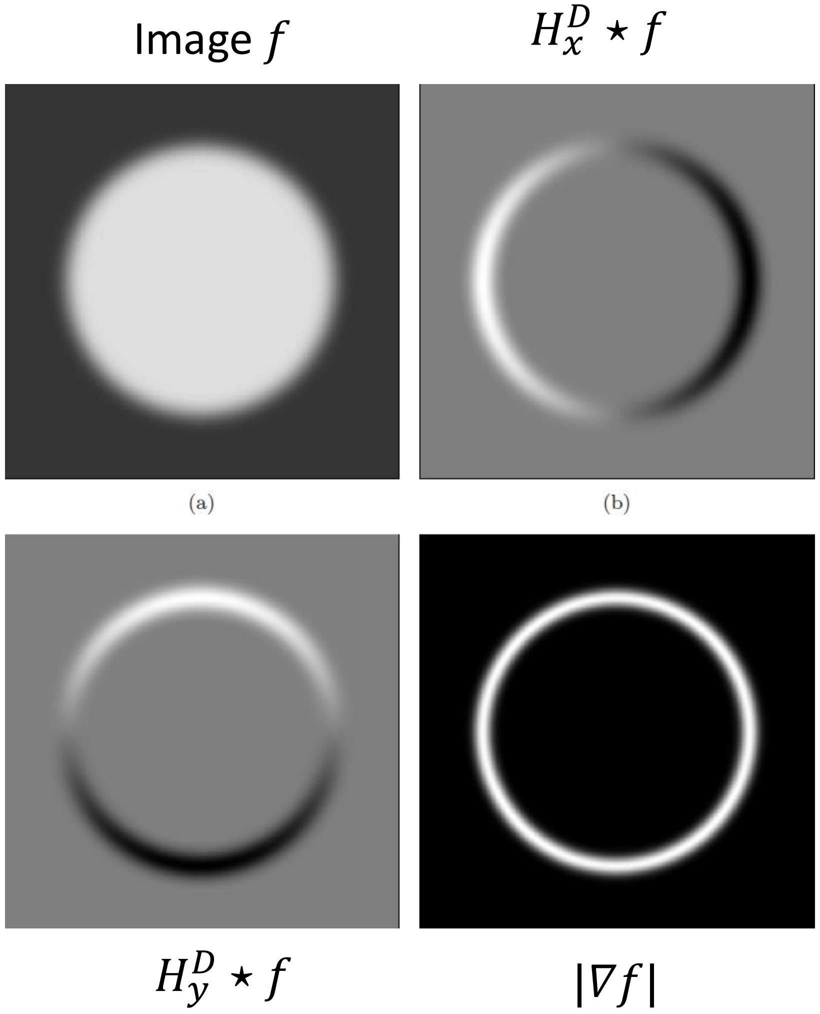
Filtres différentiels (*Derivative filters*)

- Représentation du gradient par son **approximation en différence finie**
- Filtre linéaire pour la composante x du gradient

$$H_x^D = [-0.5 \quad \mathbf{0} \quad 0.5]$$

- Filtre linéaire pour la composante y du gradient

$$H_y^D = \begin{bmatrix} -0.5 \\ \mathbf{0} \\ 0.5 \end{bmatrix}$$



Effet du bruit

- **Où se situe la bordure ?**

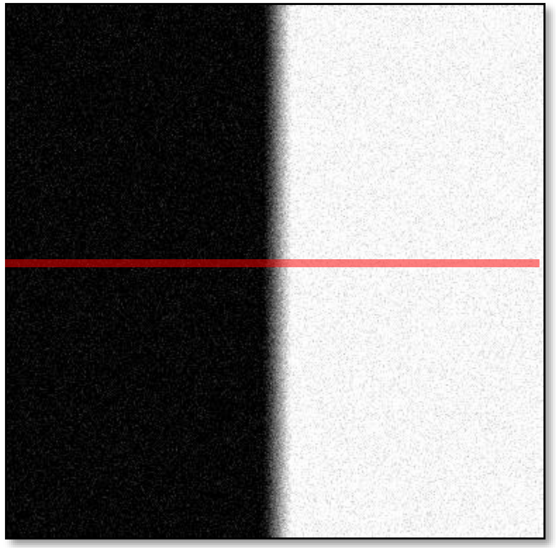
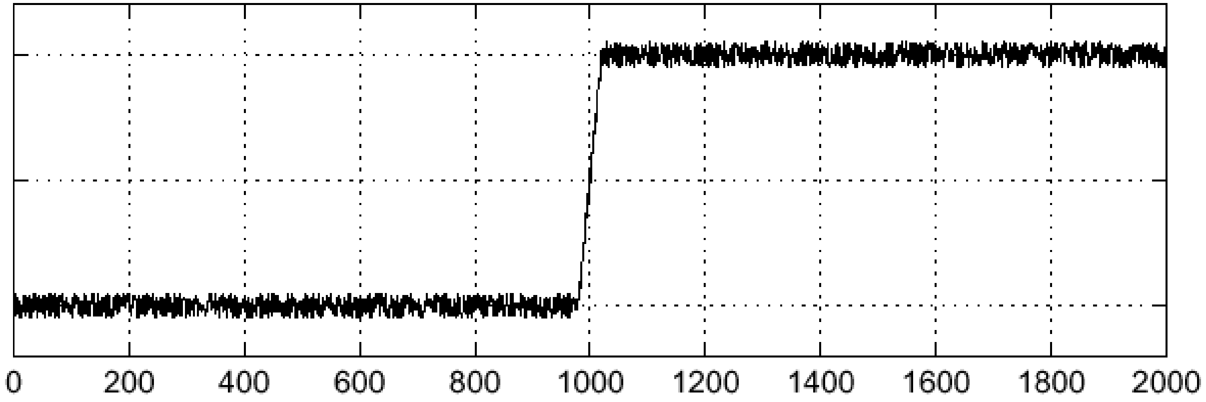
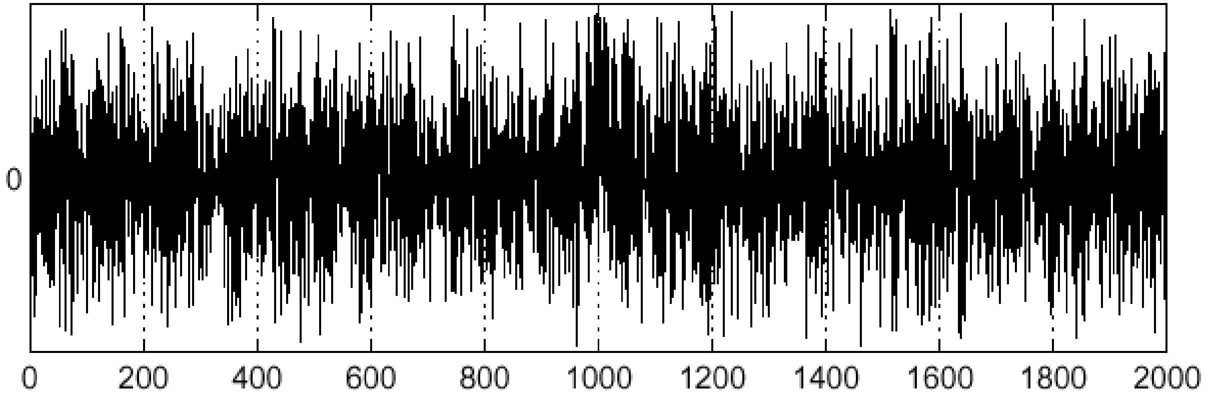


Image bruitée

$$f(x)$$



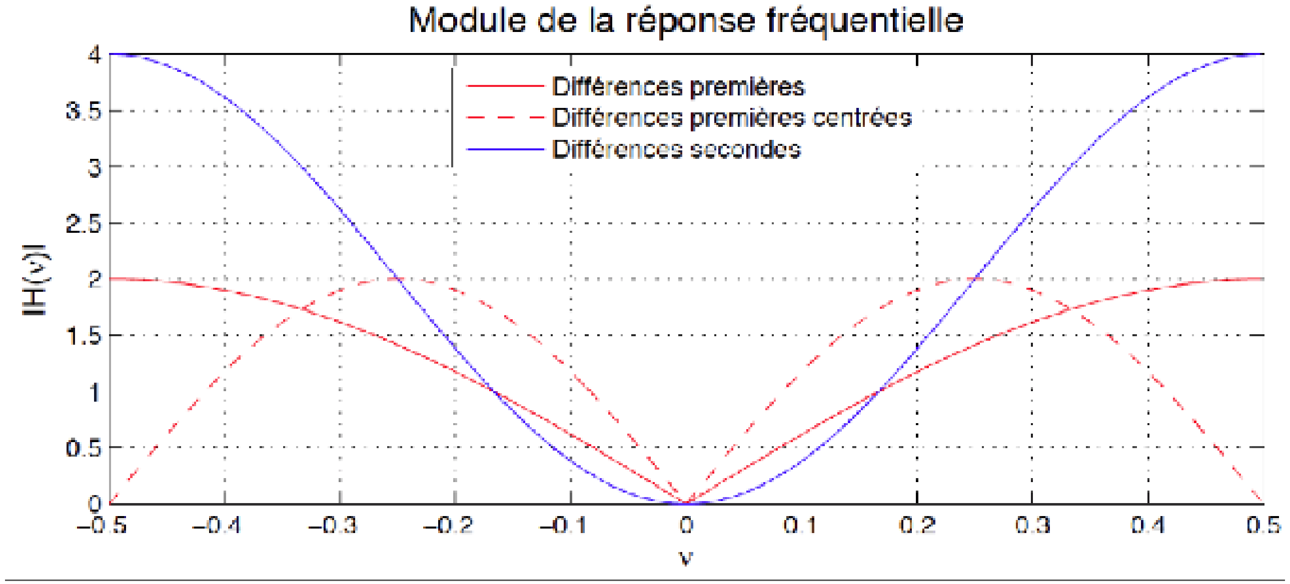
$$\frac{d}{dx}f(x)$$



Source: S. Seitz

Effet du bruit : Hautes fréquences

- Le **bruit** contient des **hautes fréquences**
- Les bordures contiennent aussi des hautes fréquences
- La différentiation est un filtre passe-haut
- **La différentiation accentue le bruit**



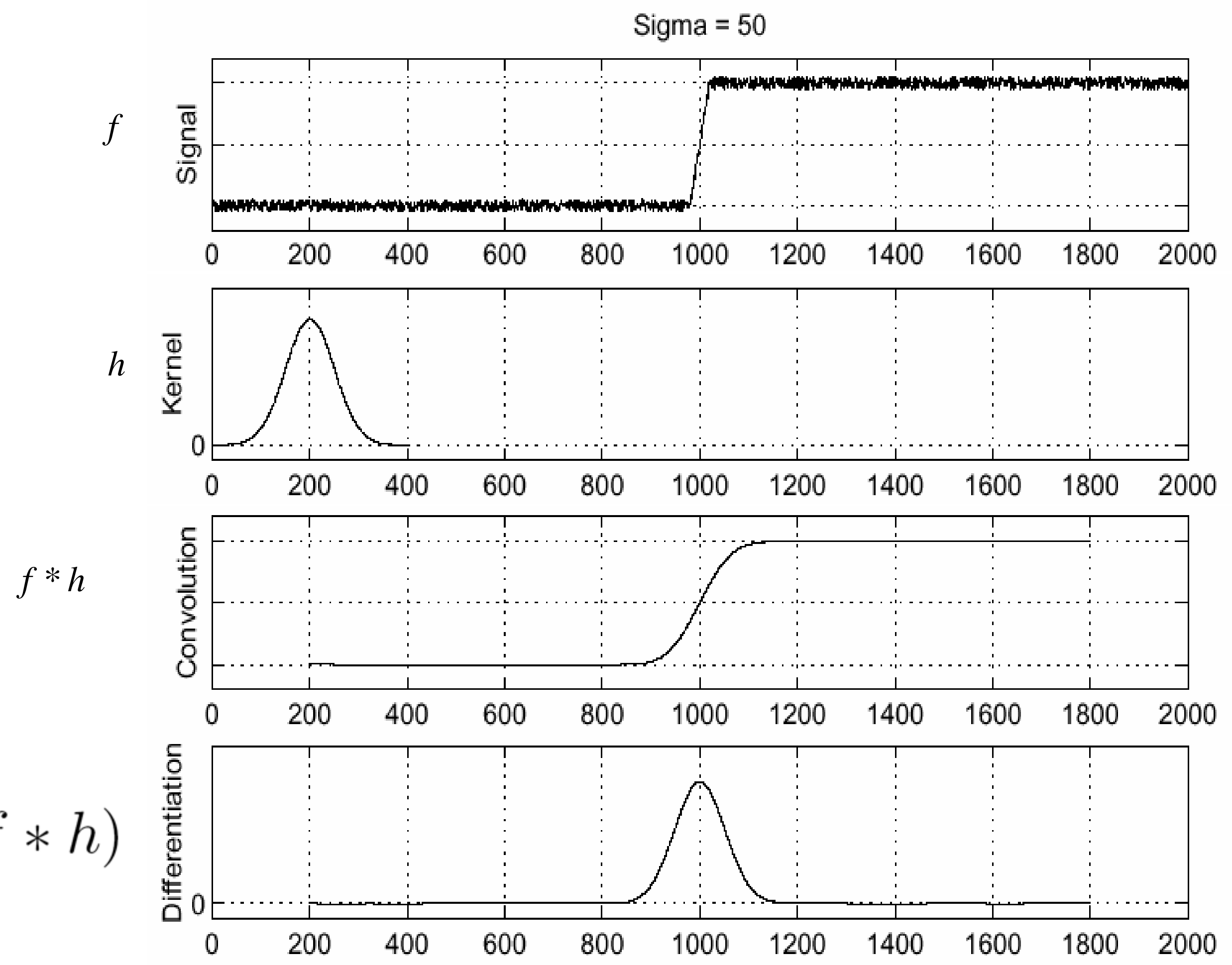
Solution : Appliquer un lissage

- Pour détecter les contours, chercher les pics dans la dérivée partielle d'une **version lissée** de l'image originale

$$\frac{\partial}{\partial x}(f * h)$$

*où h est **un filtre passe-bas** (exemple : Filtre gaussien)*

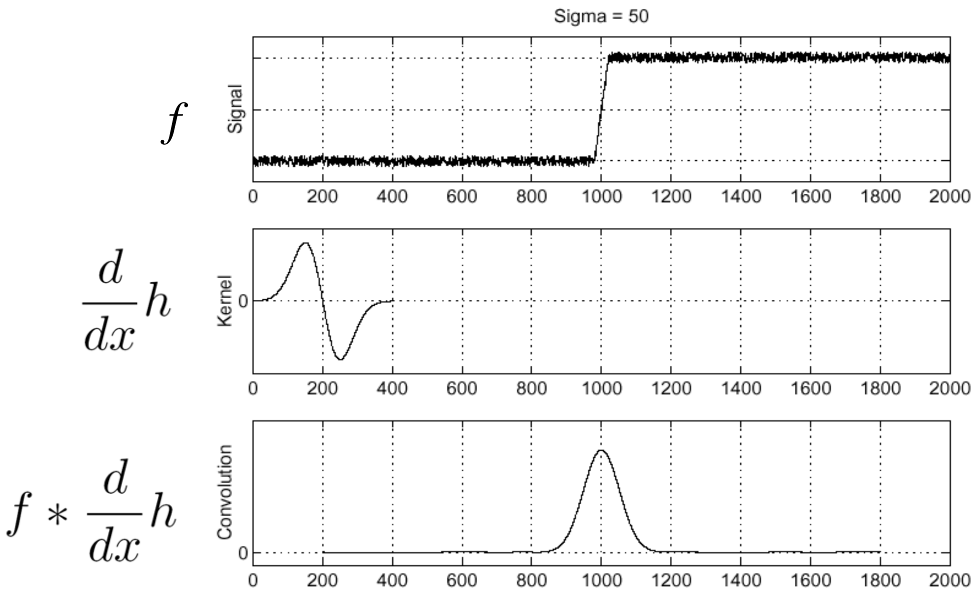
$$\frac{d}{dx}(f * h)$$



Source: S. Seitz

Propriété d'associativité de la convolution

- La différentiation est une convolution
- La convolution est associative : $\frac{d}{dx}(f * h) = f * \frac{d}{dx}h$
- Ceci permet de réduire le nombre d'opérations



Source: S. Seitz

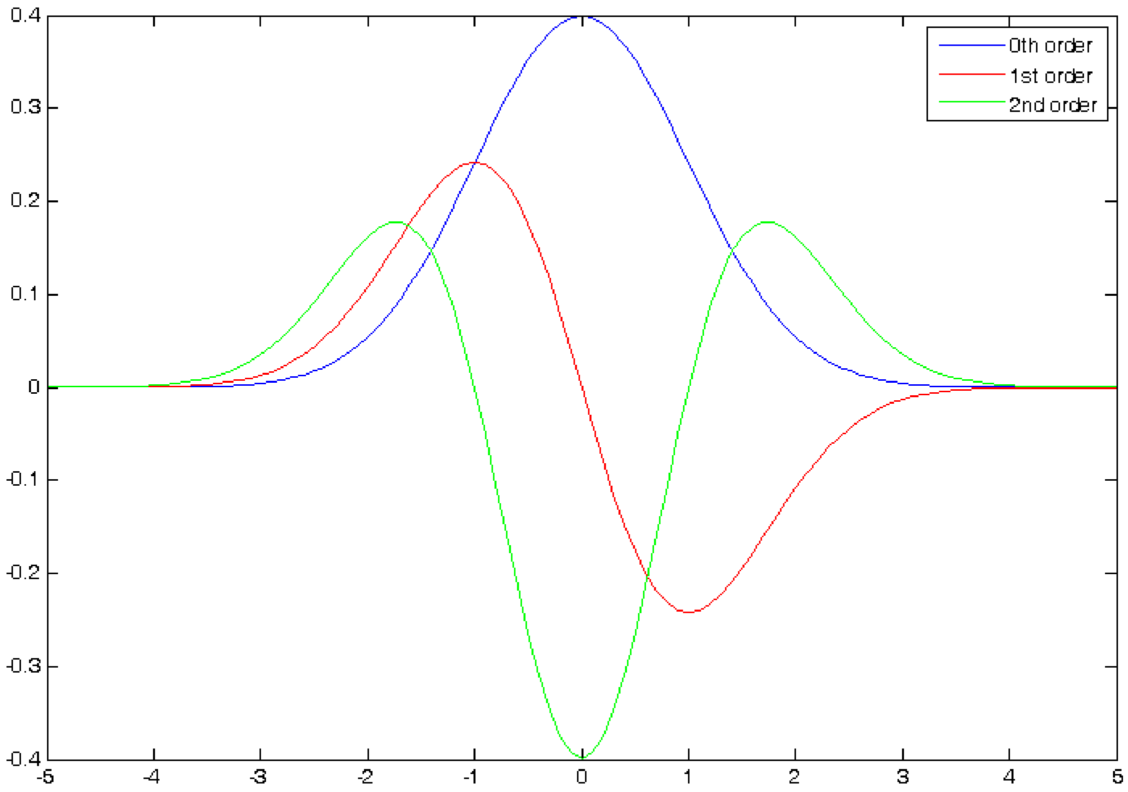
Rappel : Filtre gaussien 1D & dérivées

- Filtre gaussien

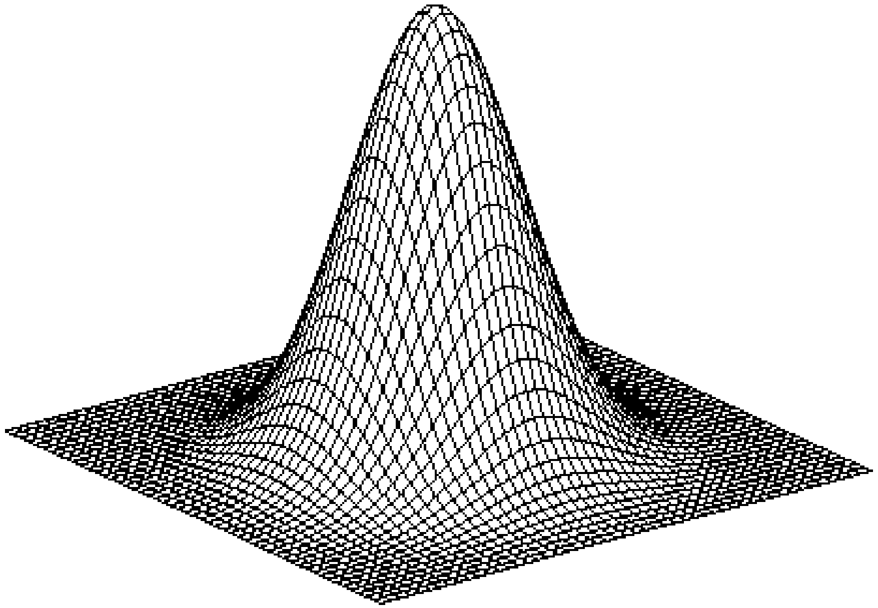
$$G_\sigma = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}$$

- 1ere dérivée du filtre gaussien

$$G'_\sigma(x) = \frac{d}{dx} G_\sigma(x)$$
$$= -\frac{1}{\sigma}\left(\frac{x}{\sigma}\right) G_\sigma(x)$$

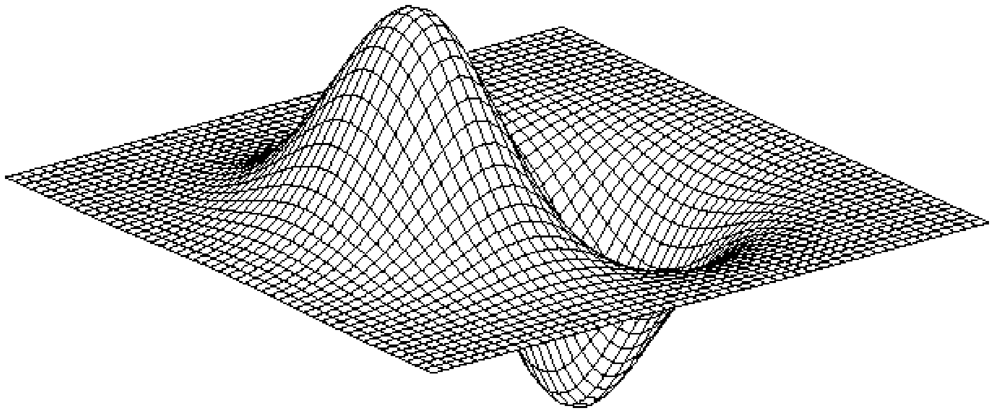


Filtre pour la détection des contours 2D



Gaussienne

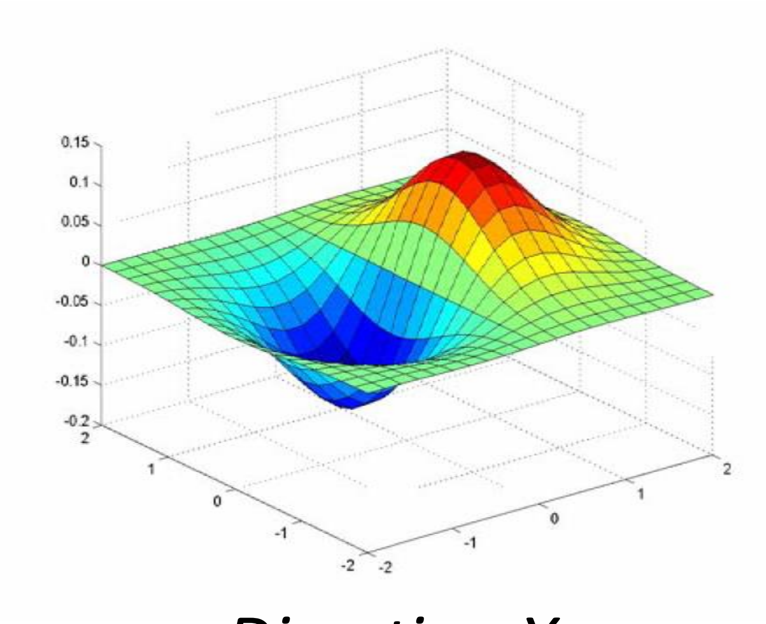
$$h_\sigma(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$



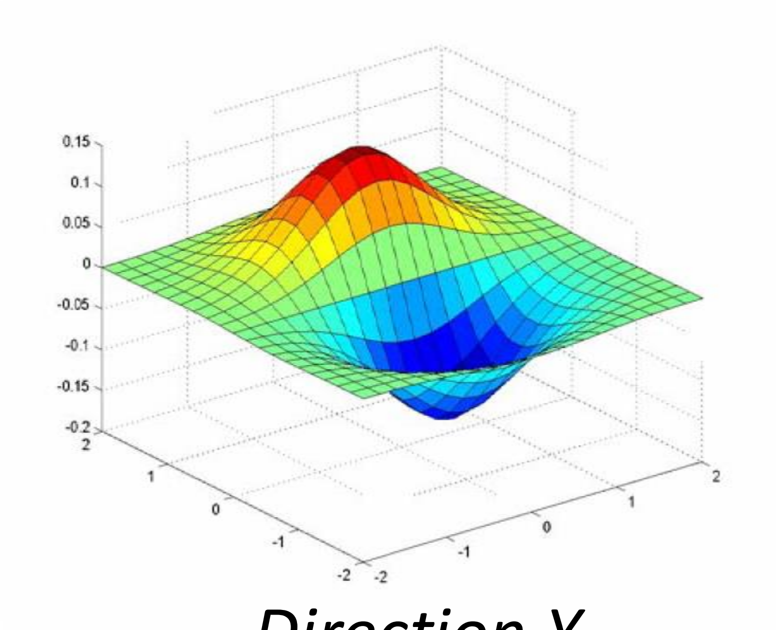
Dérivée partielle x de la Gausienne

$$\frac{\partial}{\partial x} h_\sigma(x, y)$$

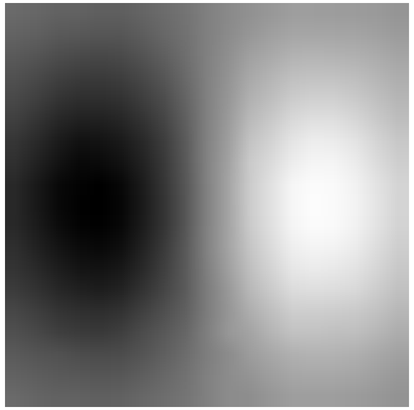
Dérivées partielles du filtre gaussien

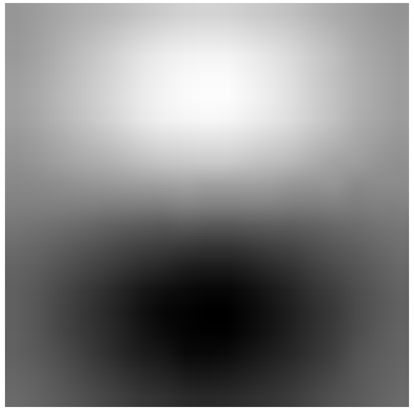


Direction X



Direction Y



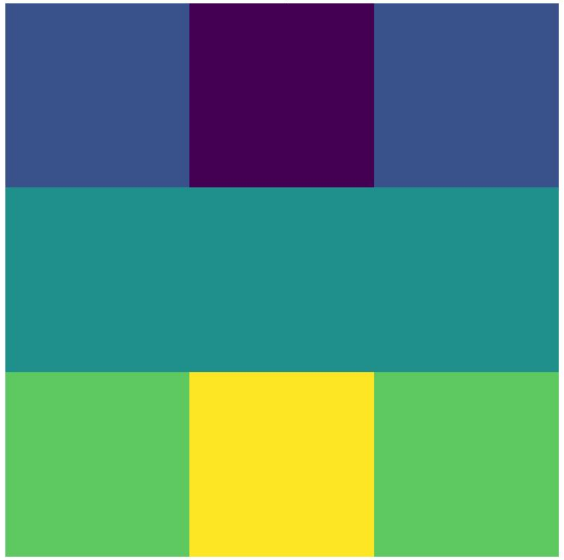


Opérateur de Sobel

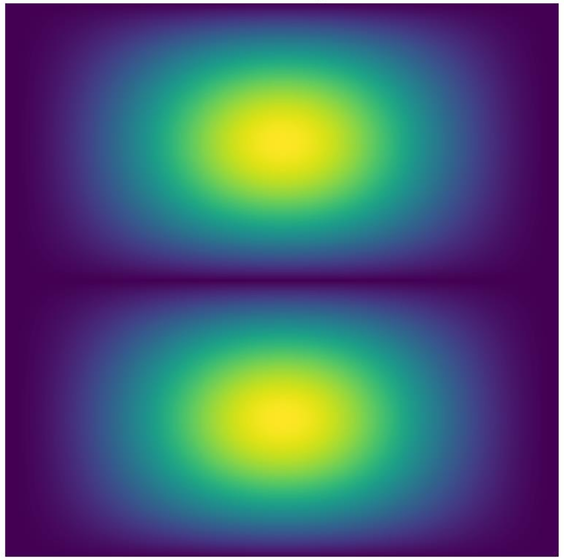
- Approximation de la dérivée de la gaussienne

$$H_x^S = \begin{bmatrix} -1 & 0 & 1 \\ -2 & \mathbf{0} & 2 \\ -1 & 0 & 1 \end{bmatrix}, H_y^S = \begin{bmatrix} -1 & -2 & -1 \\ 0 & \mathbf{0} & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

H_y^S



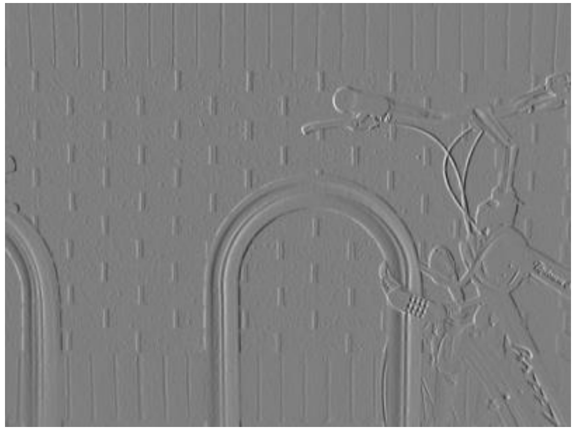
$|\mathrm{FFT}(H_y^S)|$

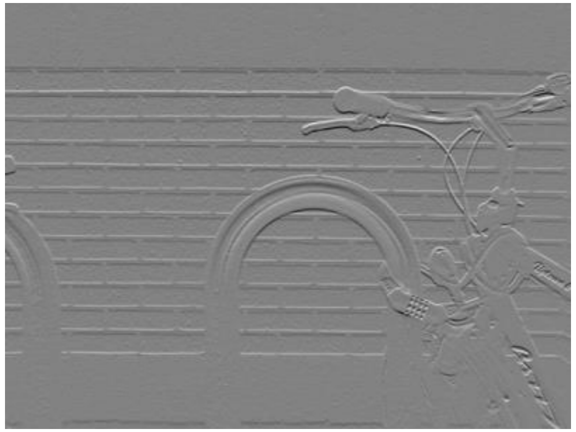


Exemple : Opérateur de Sobel

Image originale f

$G_x = H_x^s * f$





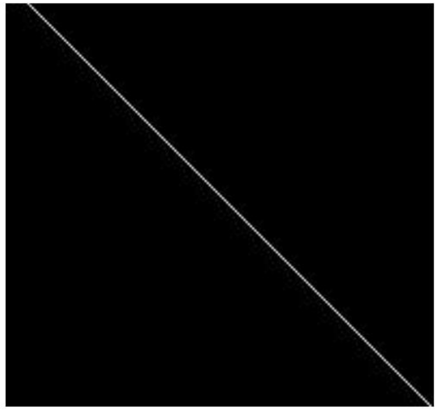
$G_y = H_y^s * f$

Amplitude $\sqrt{G_x^2 + G_{y_2}}$

Source: Wikipedia

Exemple : Sobel + Bruit

Image with Edge



Edge Location

Source : N. Snavely

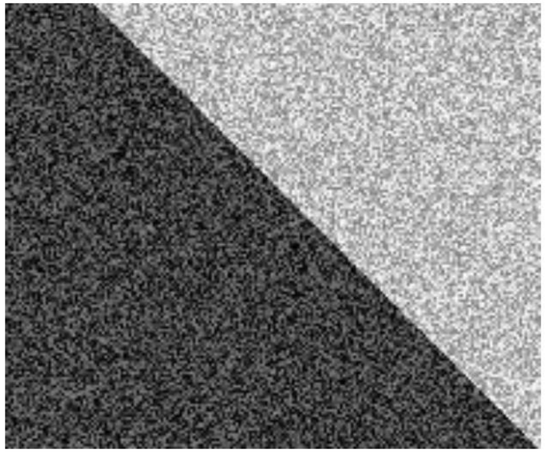
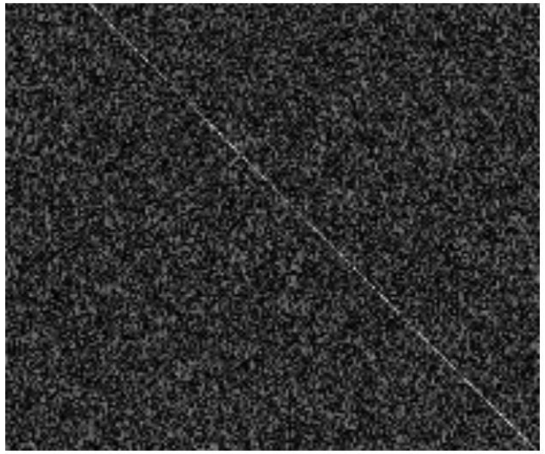
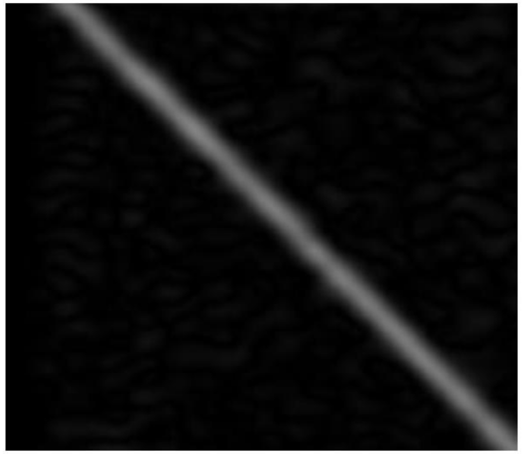


Image + Noise



Derivatives detect edge *and* noise



Smoothed derivative removes noise, but blurs edge

Opérateurs de Sobel & Prewitt

- Lissage combiné au calcul de la différence finie
- Opérateur de **Prewitt** : lissage uniforme

$$H_x^P = \begin{bmatrix} 1 \\ \mathbf{1} \\ 1 \end{bmatrix} * \begin{bmatrix} -1 & \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 1 \\ -1 & \mathbf{0} & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

- Opérateur de **Sobel** : Lissage gaussien

$$H_x^P = \begin{bmatrix} 1 \\ \mathbf{2} \\ 1 \end{bmatrix} * \begin{bmatrix} -1 & \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & \mathbf{0} & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Approximation du gradient de l’image avec les opérateurs de Prewitt & Sobel

- Approximation du gradient avec l’opérateur de Prewitt

$$\nabla I(x,y) \approx \frac{1}{6}\begin{bmatrix}(I * H_x^P)(x,y)\\ (I * H_y^P)(x,y)\end{bmatrix}$$

- Approximation du gradient avec l’opérateur de Sobel

$$\nabla I(x,y) \approx \frac{1}{8}\begin{bmatrix}(I * H_x^S)(x,y)\\ (I * H_y^S)(x,y)\end{bmatrix}$$

Orientation et force des contours pour Sobel / Prewitt

- Calcul des **composantes X et Y** du gradient

$$D_x = H_x * I \text{ et } D_y = H_y * I$$

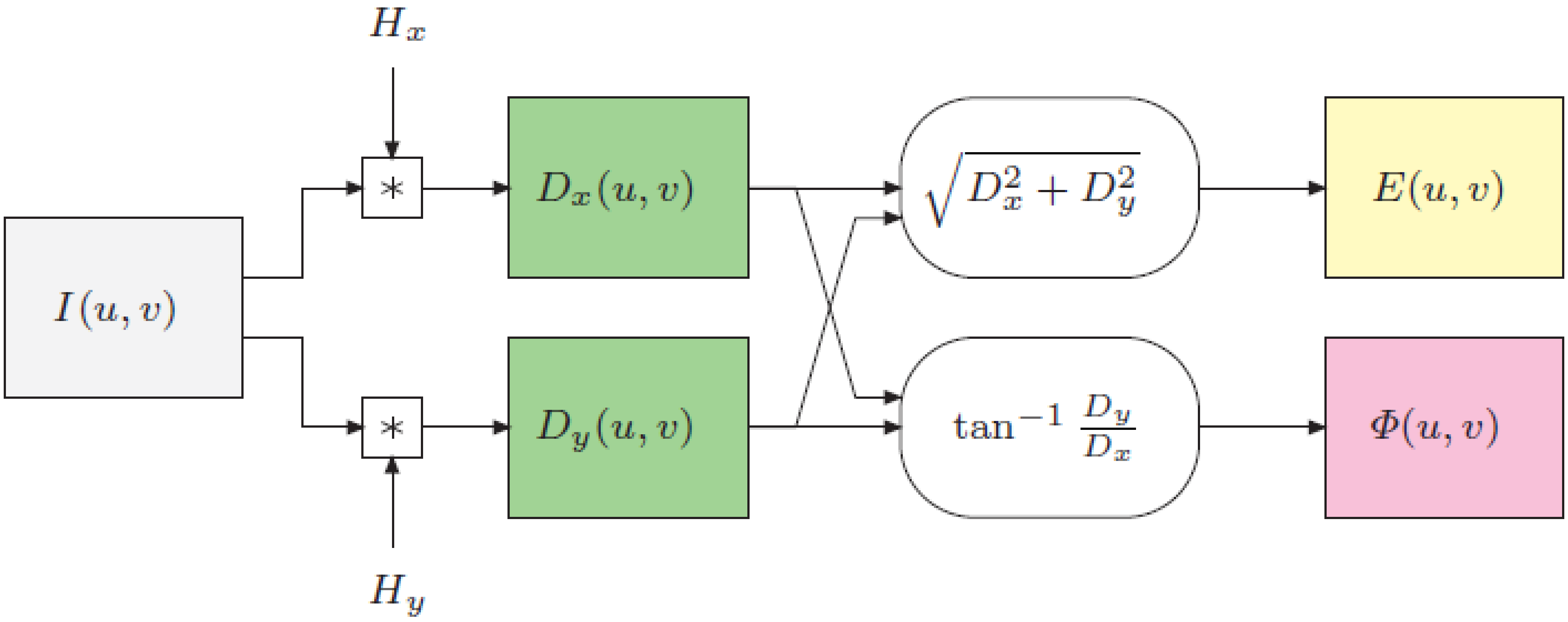
- **Force du contour**

$$E(u,v) = \sqrt{(D_x(x,y))^2 + (D_y(x,y))^2}$$

- **Orientation du contour**

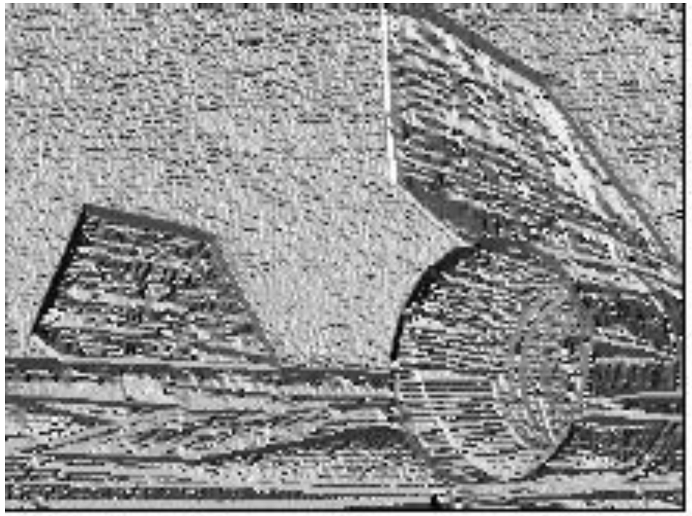
$$\Phi(x,y) = \tan^{-1}\left(\frac{D_y(u,v)}{D_x(u,v)}\right)$$

Représentation graphique du pipeline d'analyse des contours avec Sobel



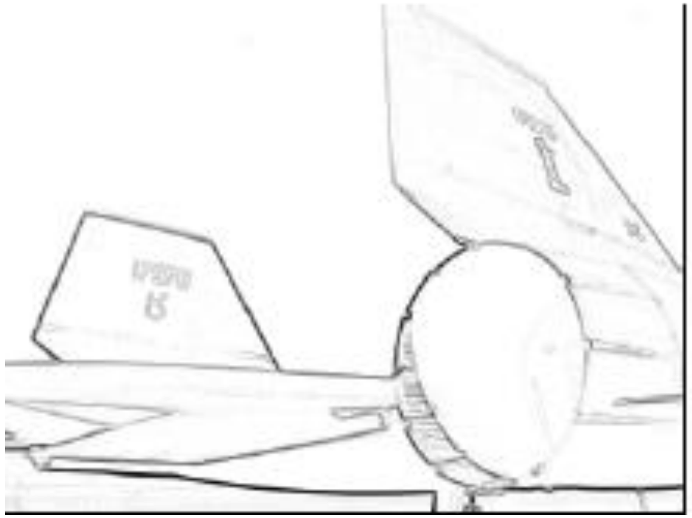
Exemple : Force & Orientation des contours avec Sobel

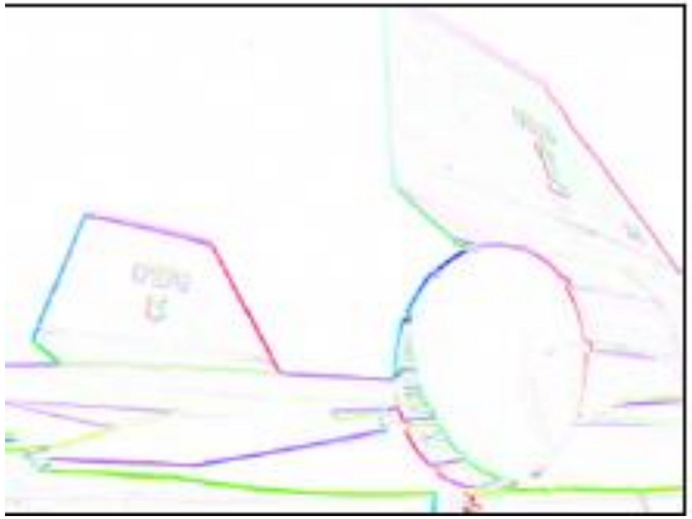
Image originale



Orientation des contours

Force des contours





Orientation des contours
(en couleur)

Détection des contours : Autre opérateur

- On peut utiliser le **Laplacien** (2^e dérivée) plutôt que le gradient (1^{ere} dérivée) pour détecter les contours.

$$\nabla^2 f(x,y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

- **Amplitude** : rapidité des variations de f
- **Direction** : plus grande pente
- **Approximations numériques**

$$\boldsymbol{\nabla^2} = \begin{pmatrix} \mathbf{0} & \mathbf{-1} & \mathbf{0} \\ \mathbf{-1} & \mathbf{4} & \mathbf{-1} \\ \mathbf{0} & \mathbf{-1} & \mathbf{0} \end{pmatrix}$$

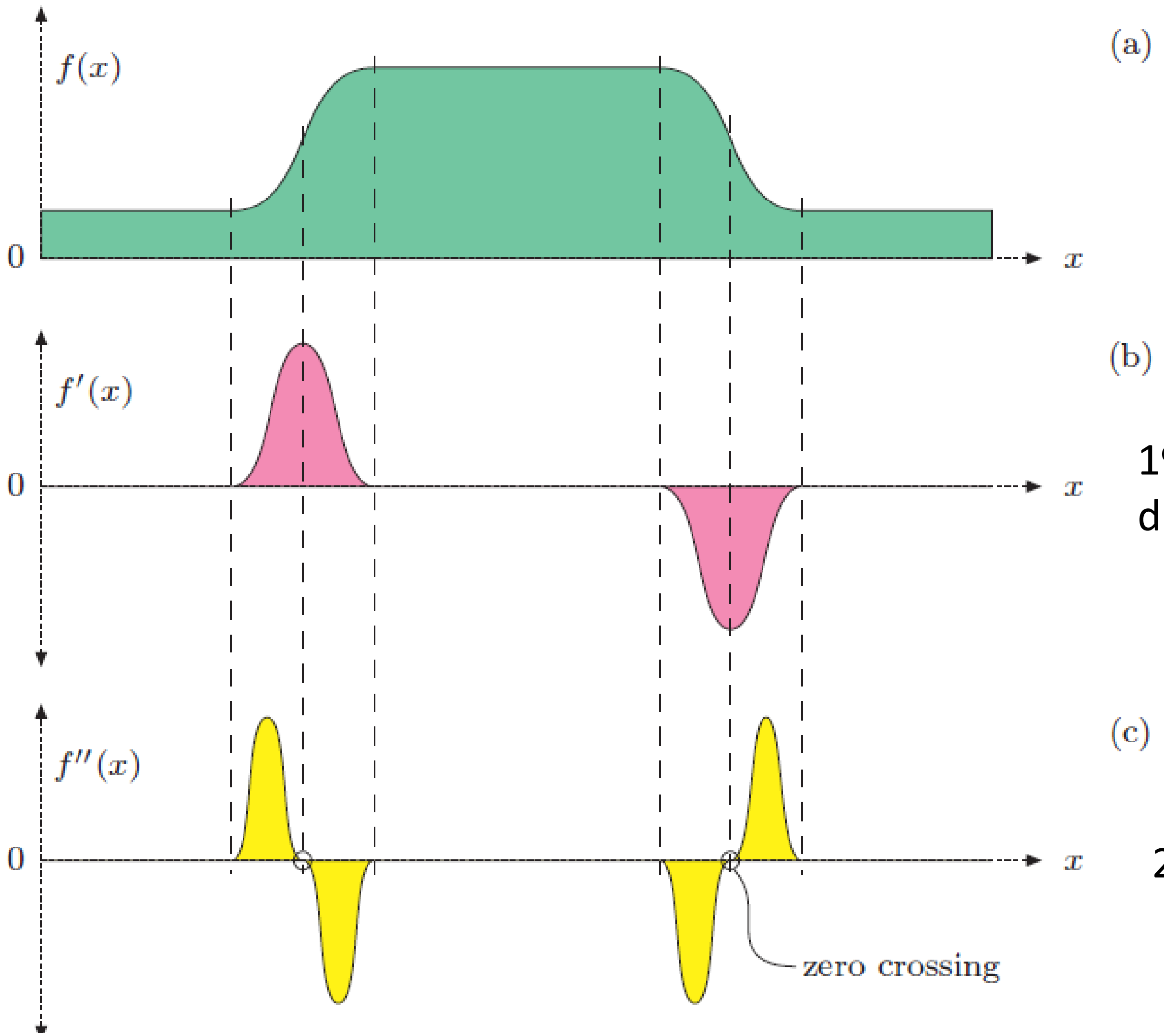
Sommes d'opérations 1D

$$\boldsymbol{\nabla^2} = \begin{pmatrix} \mathbf{-1} & \mathbf{-1} & \mathbf{-1} \\ \mathbf{-1} & \mathbf{8} & \mathbf{-1} \\ \mathbf{-1} & \mathbf{-1} & \mathbf{-1} \end{pmatrix}$$

Meilleure isotropie

Comparaison entre 1ere / 2e dérivée

(Burger, 2009)

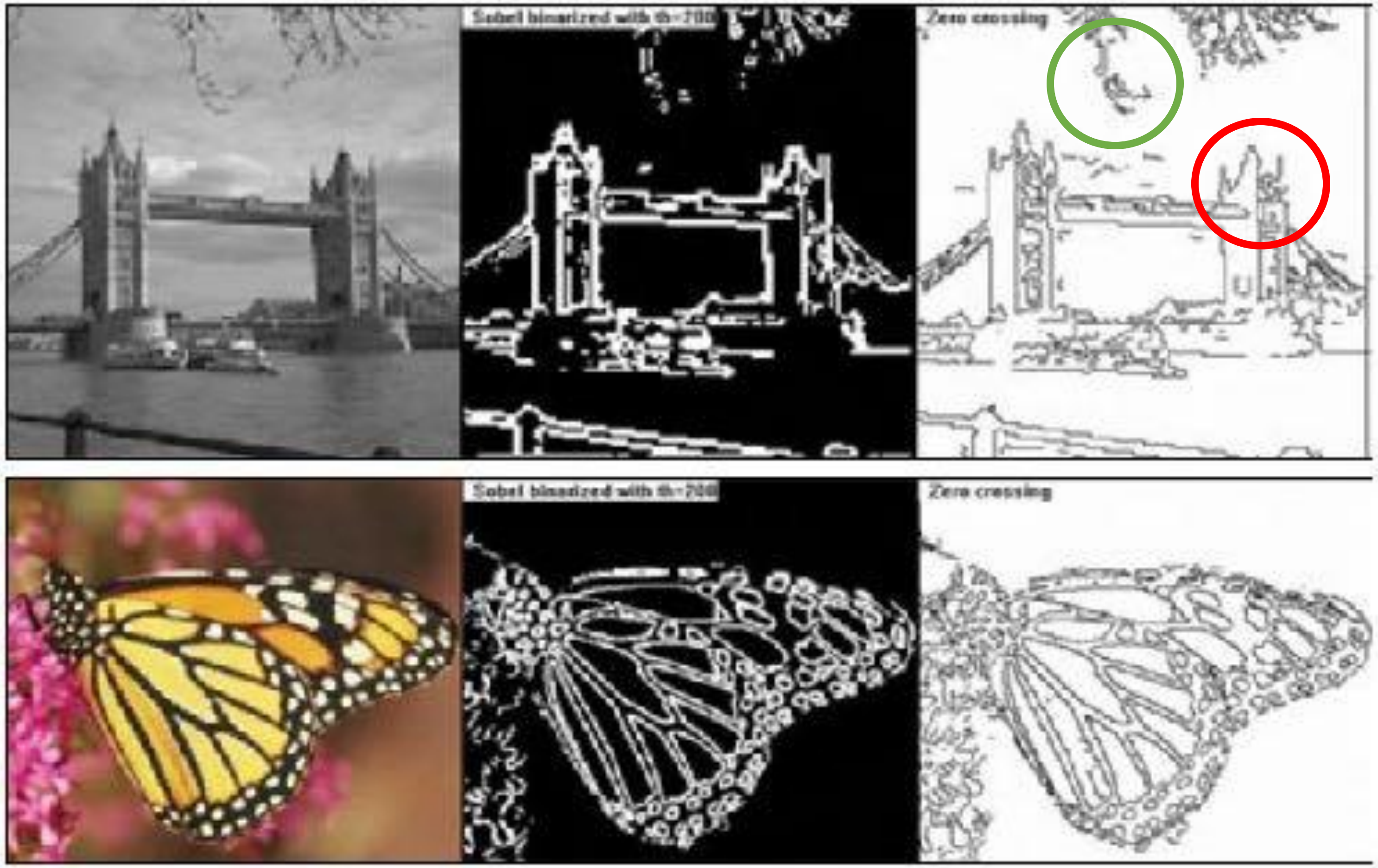


1ere dérivée

2e dérivée

Comparaison : Gradient vs Laplacien

Image | Sobel | Marr-Hildreth



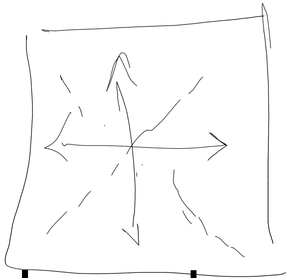
Description plus fine

Manque certains détails

Synthèse sur les méthodes élémentaires de détection de contours

Gradient

- Directionnel
- Moins sensible au bruit (différences premières)
- Possibilité de mise en œuvre par opérateurs 1D



Laplacien

- Isotrope → moins sensible aux diag
- Très sensible au bruit (différences secondes)
- Possibilité limitée de mise en œuvre par opérateurs 1D
- Double réponse aux discontinuités → détection des passages par zéro

Détection des frontières des objets

- Les résultats obtenus avec les **méthodes simples** de détection des contours (p. ex. Sobel) **sont souvent différents** de ce que les **humains perçoivent** comme étant des contours importants.
- **Raison 1 :** Les opérateurs sont locaux et petits (3x3) et détectent uniquement les variations locales d'intensité.
- **Raison 2 :** Les contours existent à plusieurs échelles

Quelles sont les frontières perçues par les humains ?

- Base de données de segmentation de Berkeley ([URL](URL))

Image | Segmentation humaine | Amplitude du gradient

Source : Malik & Hays

Méthode de Canny pour détecter les contours

- Probablement la méthode de détection de contours **la plus utilisée** en vision par ordinateur
- **Modèle théorique :** Des contours échelons corrompus par du bruit gaussien additif.
- **Canny** a montré que la 1^{ere} dérivée de la gaussienne est une bonne approximation de l'opérateur permettant d'optimiser le ratio signal-sur-bruit et la localisation de la détection des contours.

J. Canny, ***A Computational Approach To Edge Detection***, IEEE Trans. Pattern Analysis and Machine Intelligence, 8:679-714, 1986. (> 30K citations!)

Méthode de Canny : Image initiale

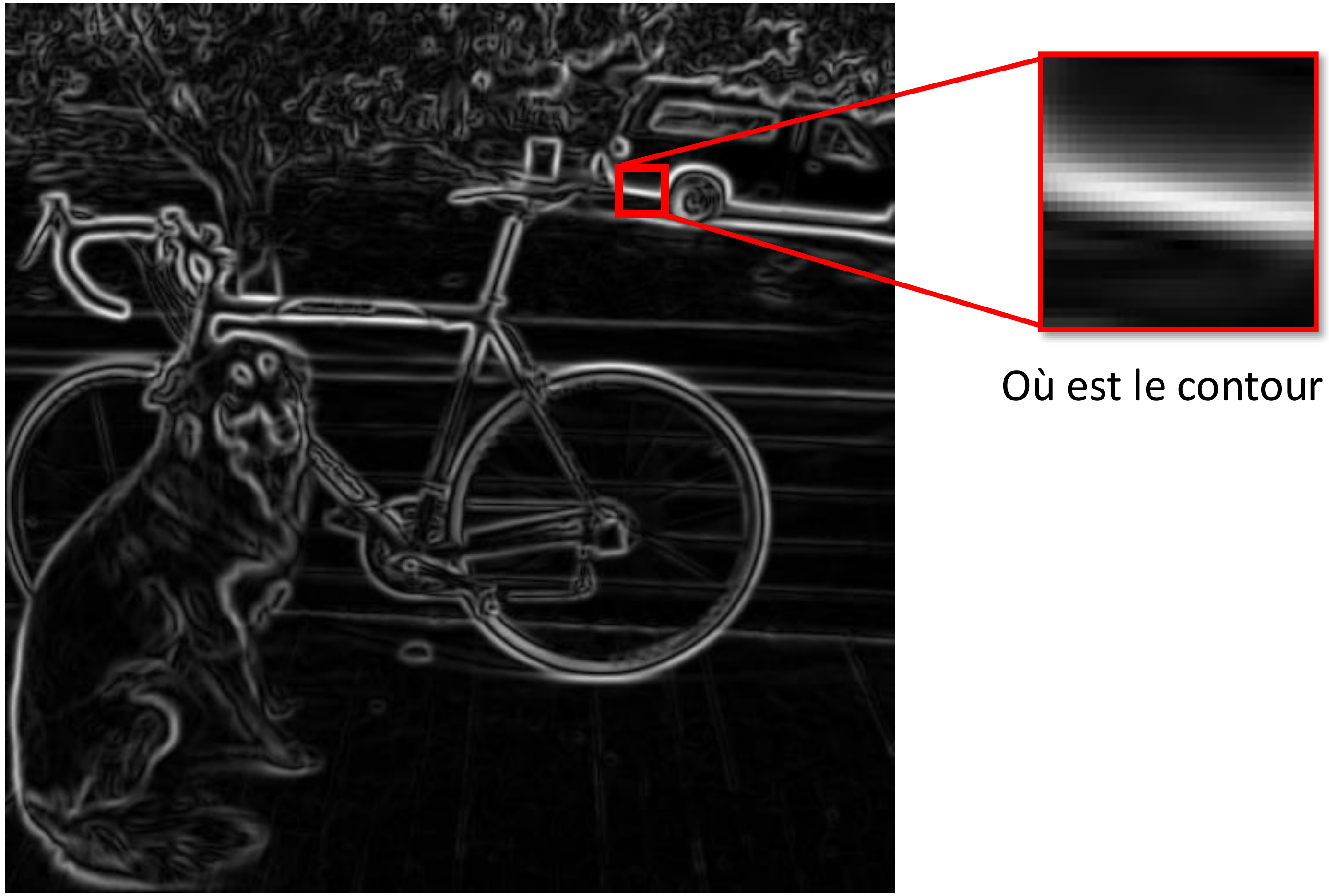
- **Démo** : http://bigwww.epfl.ch/demo/ip/demos/edgeDetector/

Source : N. Snavely

Image credit: Joseph Redmon

Méthode de Canny – Algorithme (1)

1. Filtrer l'image avec les dérivées x et y de la gaussienne

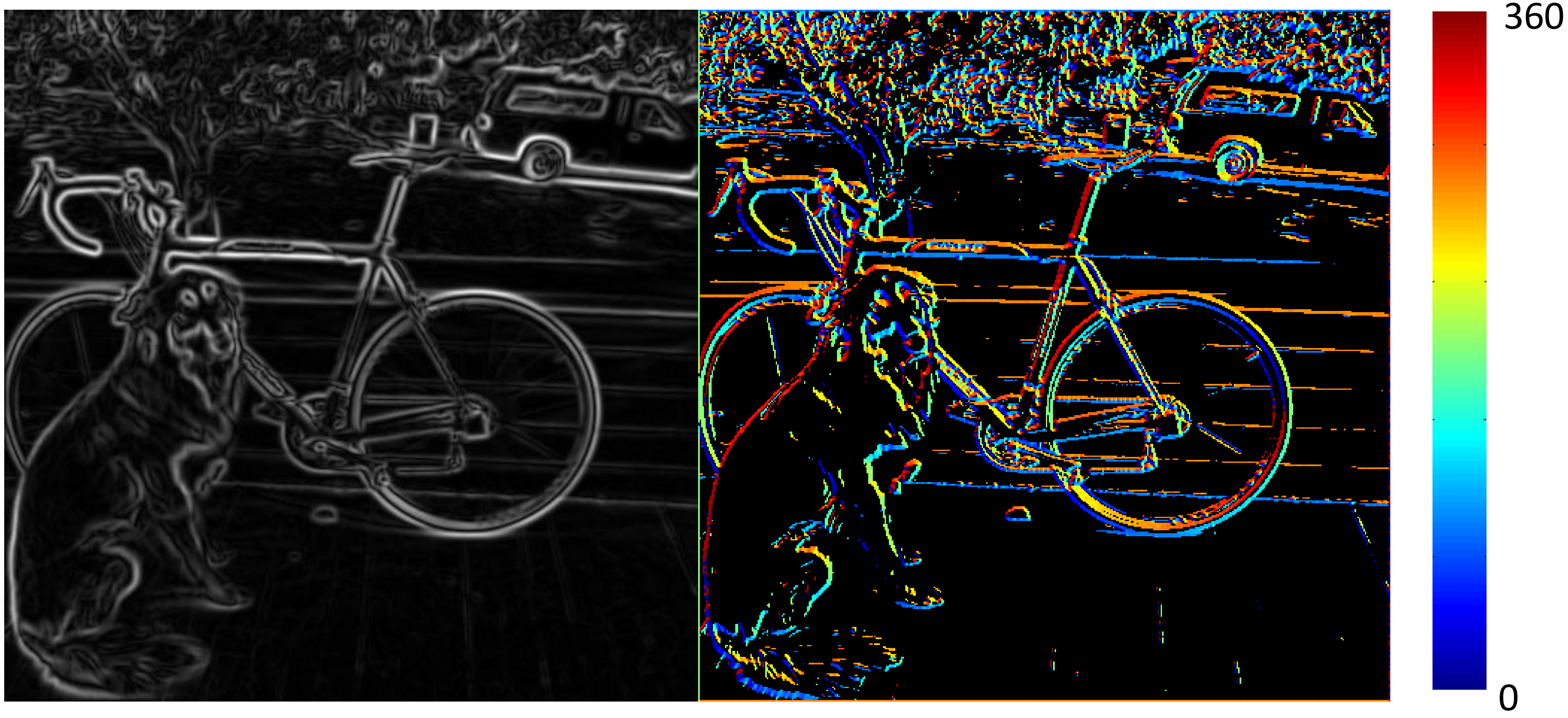


Amplitude du gradient

Où est le contour ?

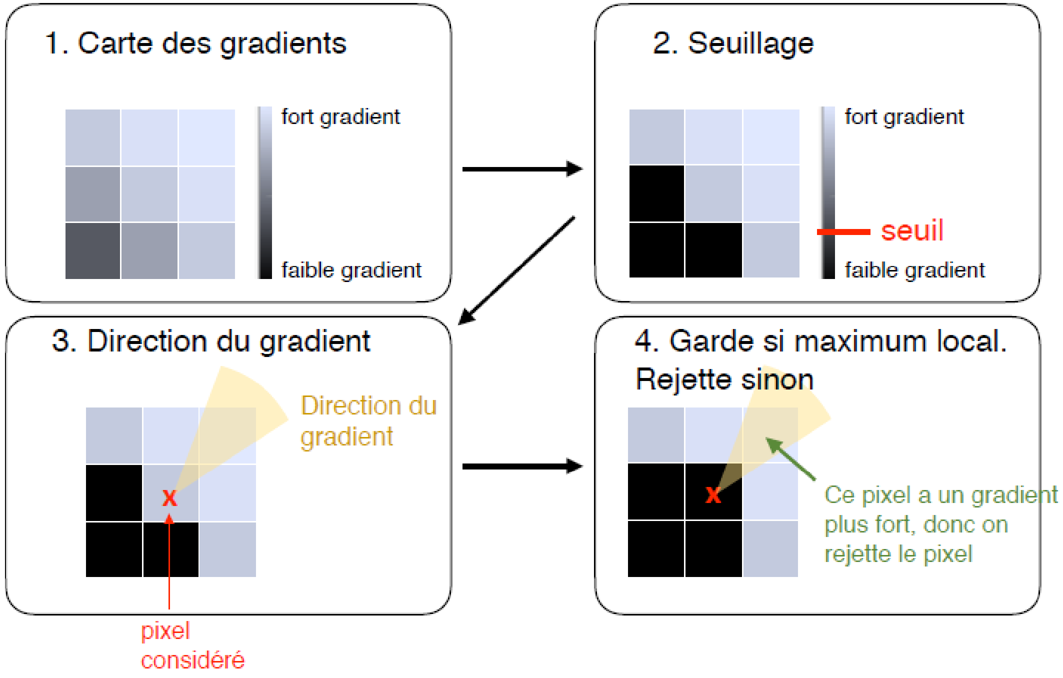
Méthode de Canny – Algorithme (2)

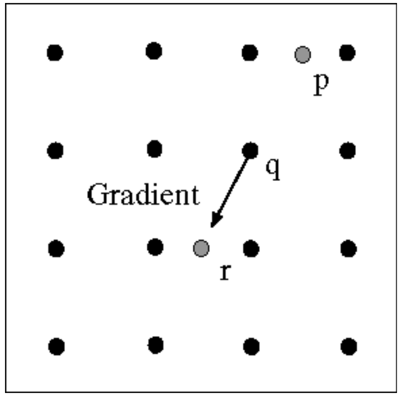
2. Calcul de l’amplitude et de l’orientation du gradient.

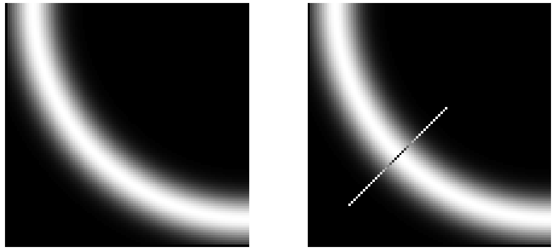


Méthode de Canny – Algorithme (3)

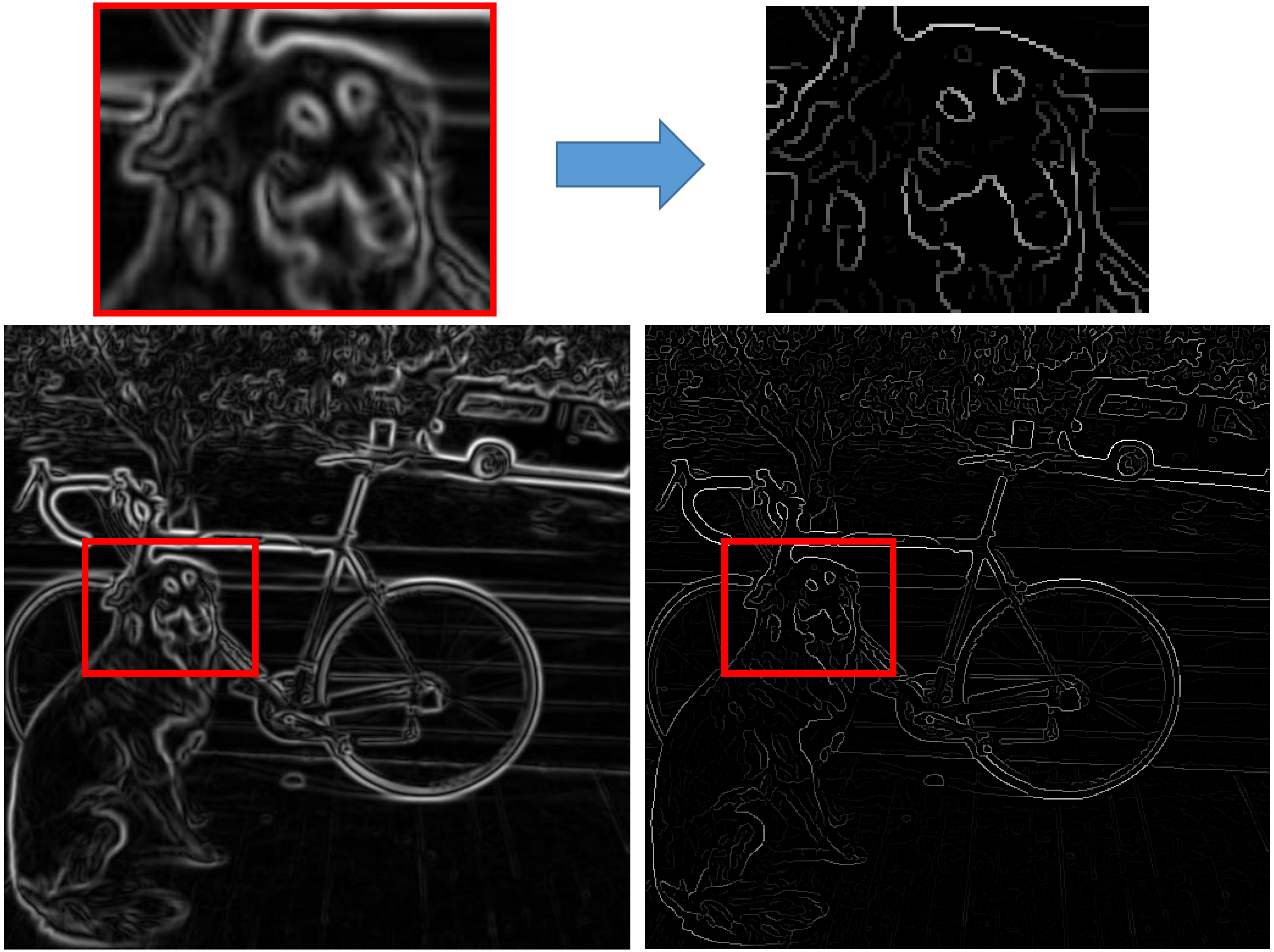
3. Affinage des contours via la suppression des non-maximums
   - Convertir les arêtes minces d'une largeur de plusieurs pixels en une ligne d'une largeur d'un seul pixel





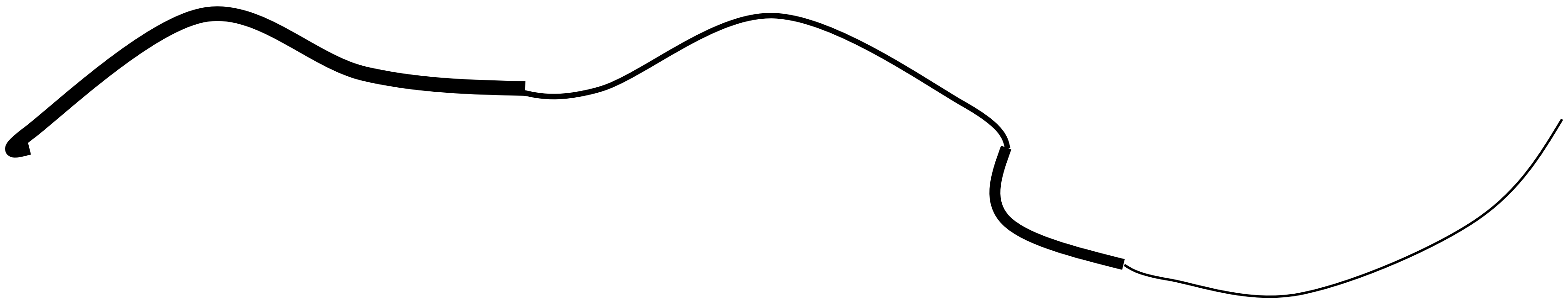


Exemple : Suppression des non-maximums



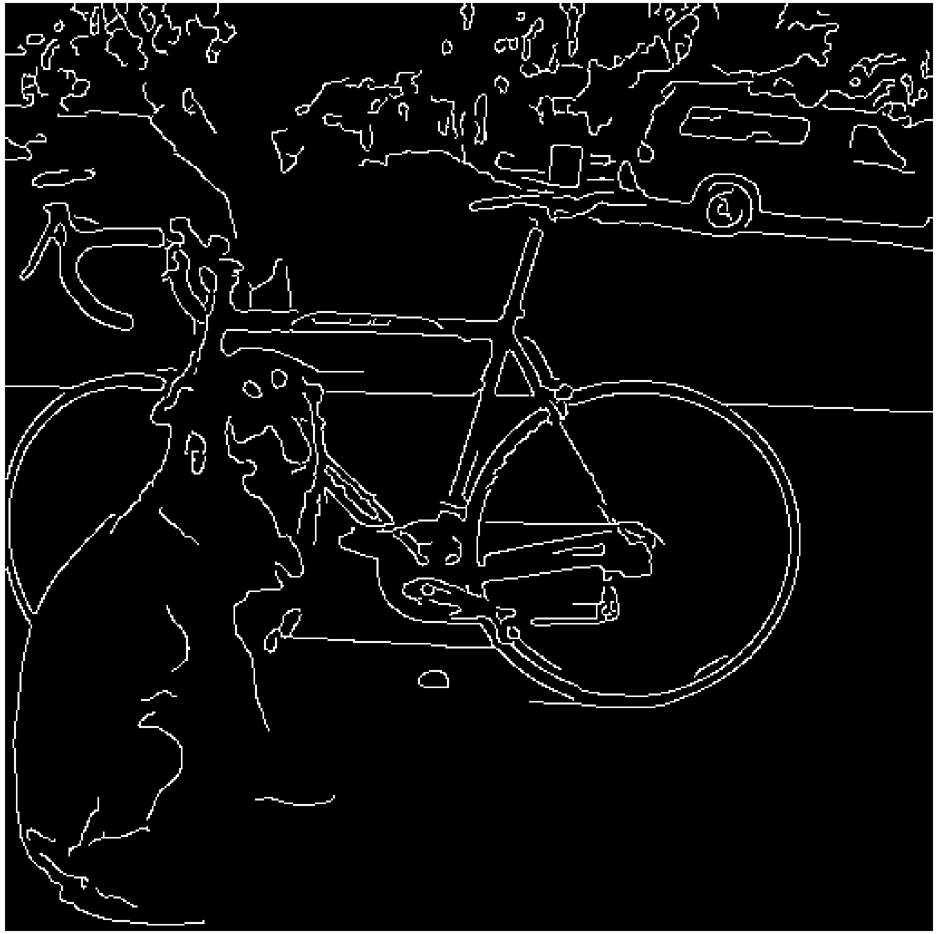
Méthode de Canny – Algorithme (4) Double seuillage (Hystérésis)

- 2 seuils : Bas (S_l) et élevé (S_h)
- Amplitude du gradient > S_h? = Contour fort
- Amplitude du gradient < S_l ? = bruit
- Entre les deux : Contour faible
- « Suivi » des contours en débutant sur les pixels de contours forts
- Étendre les contours forts vers les contours faibles



Source: S. Seitz

Exemple : Seuillage des contours



Résumé : Méthode de Canny

Étapes principales du filtre de Canny

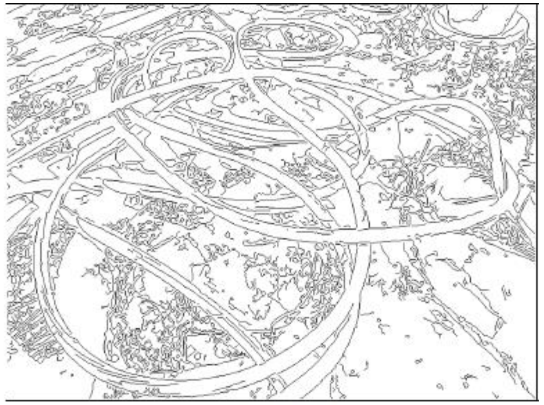
1. **Filtrer l'image** avec un filtre dérivée de la gaussienne
2. Trouver la **direction** et la **l'amplitude** du gradient
3. **Affinage des contours** (Suppression des non-maximums)
4. **Double seuil** (hystérésis) pour déterminer les arêtes potentielles, et **suivi des bordures par hystérésis**



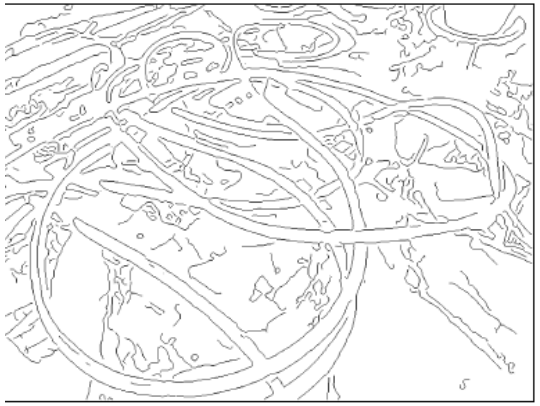

Canny - Résumé

- **1er pipeline de vision par ordinateur !**
- Encore très utilisé
- Dépend de plusieurs paramètres : seuils bas et élevé, σ : taille du filtre gaussien

Original



$\sigma = 1.0$



$\sigma = 2.0$

$\sigma = 4.0$

`skimage.feature.canny` ([Documentation])

Détection de courbes simples

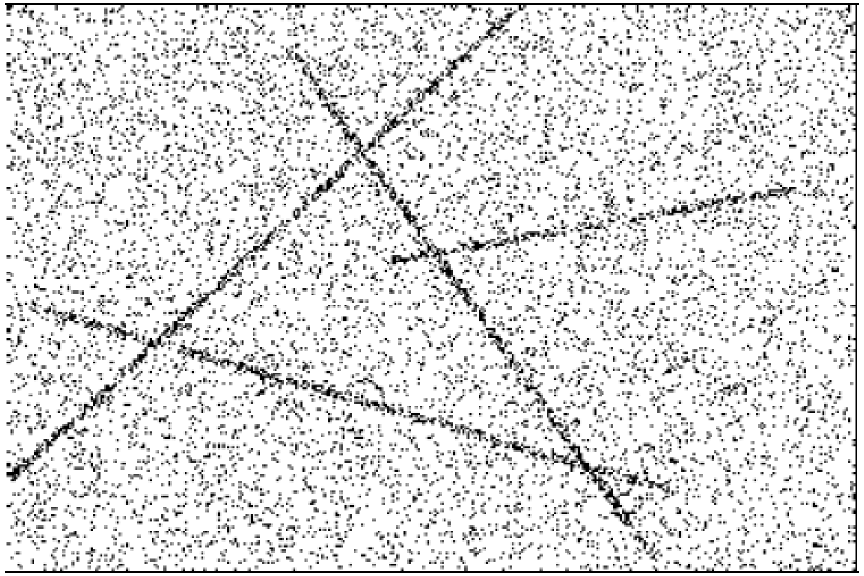
Chapitre 5 : Segmentation (Partie 1)

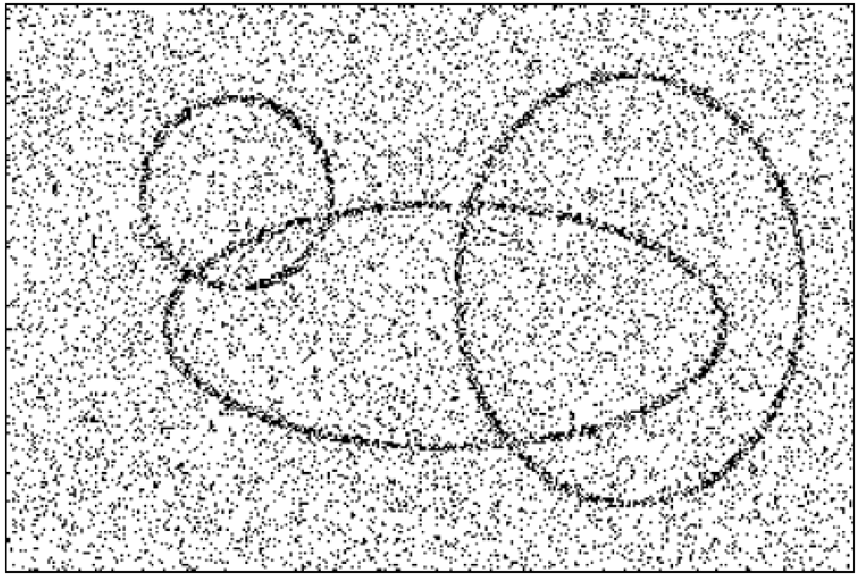
Joël Lefebvre (UQÀM)

INF600F – Traitement d’images

Détection de courbes

- **Contours détectés :** représentation **incomplète** des frontières présentes dans l'image
- Nécessité de joindre les éléments d'une même frontière
- Très difficile lorsque le niveau de bruit est élevé





(Burger, 2009)

Approches de détection de courbes

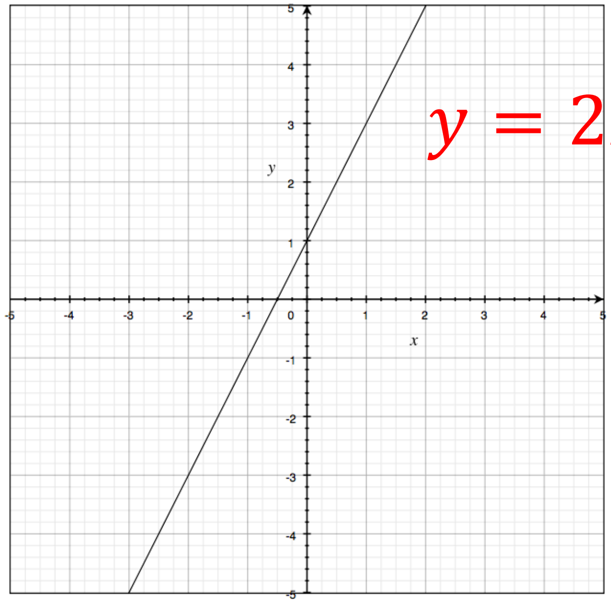
- Approche locale
  - Seuillage avec hystérésis (Canny)
  - Méthodes empiriques : amplitude similaire et même angle du gradient
  - Approche par régions (exemple : approximations polygonales)
- Approche globale : **Transformée de Hough**
  - Détection de l'ensemble des points de contour **appartenant à un même objet paramétré**
  - L'objet paramétré peut être un point, **une droite**, une courbe, un cercle, etc.

Paramétrisation d’une droite : Forme Pente / Intersection

$$y = mx + b$$

Où m est la pente et b est l’intersection avec l’axe y

Source : CMU2019



$y = 2x + 1$

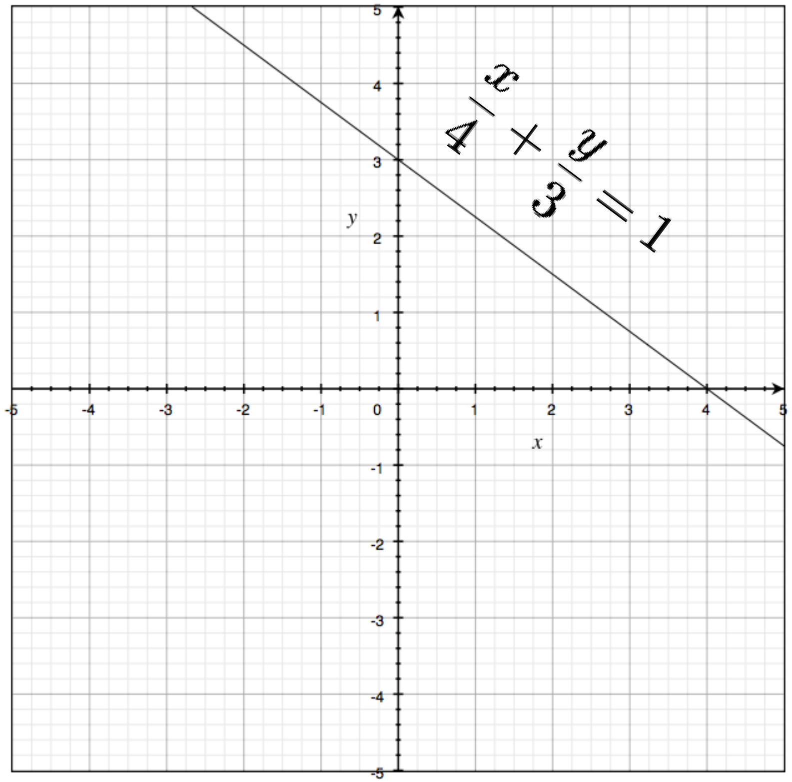
Quels sont les paramètres de cette ligne ?

Paramétrisation d'une droite : Forme Double-Intersection

$$\frac{x}{a} + \frac{y}{b} = 1$$

- a est l'intersection avec l'axe x
- b est l'intersection avec l'axe y

Quels sont les paramètres de cette ligne?



Source : CMU2019

$(x=0, y=4)$

	$x=0$	1	2	3	4
4	2	0	0	0	0
3		1	0	0	0
2	0	0	1	0	0
$y=1$	0	0	0	1	0
$y=0$	0	0	0	0	1

b \ m	-1	0	1
4	$\cancel{1}$ 2	$\cancel{0}$ 1	$\cancel{0}$ 1
3	0	0	0
2	0	0	0
1	0	0	0
0	0	0	0

$$m=-1,\ x=1,\ y=3$$
$$b=3-(-1)1=4$$

$$m=-1,\ x=0,\ y=4$$
$$\Rightarrow b=y-mx$$
$$4-(-1)0=4$$

$$m=0,\ x=0,\ y=4$$
$$b=4-0\cdot 0=4$$

$$m=1,\ X=0,\ y=4$$
$$b=4-0\cdot 1=4$$

$m = 0, \; x = 1, \; y = 3$

$b = 3 - 0 \cdot 1 = 3$

Finale

$\longrightarrow$

5	1	1
0	1	0
0	2	0
0	1	0
0	1	1

$\Rightarrow \quad y = -x + 4$

Paramétrisation d’une droite : Forme normale

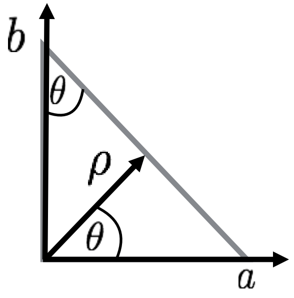
$$x \cos\theta + y \sin\theta = \rho$$

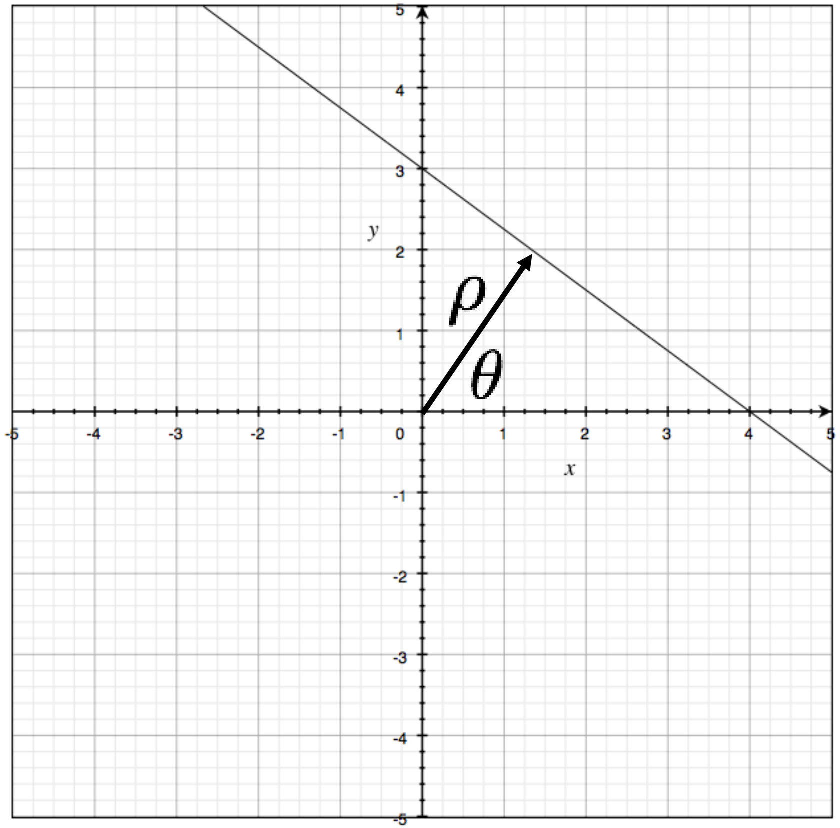
Dérivation :

$$\cos\theta = \frac{\rho}{a} \rightarrow a = \frac{\rho}{\cos\theta}$$
$$\sin\theta = \frac{\rho}{b} \rightarrow b = \frac{\rho}{\sin\theta}$$

Remplacer dans : $\frac{x}{a} + \frac{y}{b} = 1$

On obtient : $x \cos\theta + y \sin\theta = \rho$





Transformée de Hough

Source : CMU2019

- Détection de l’ensemble des points de contour **appartenant à un même objet paramétré**
- L’objet paramétré peut être un point, une droite, une courbe, un cercle, etc.
- Les contours n’ont pas besoin d’être connectés
- Les droites peuvent être partiellement cachées
- Principe général : les contours **votent** pour les modèles possibles.

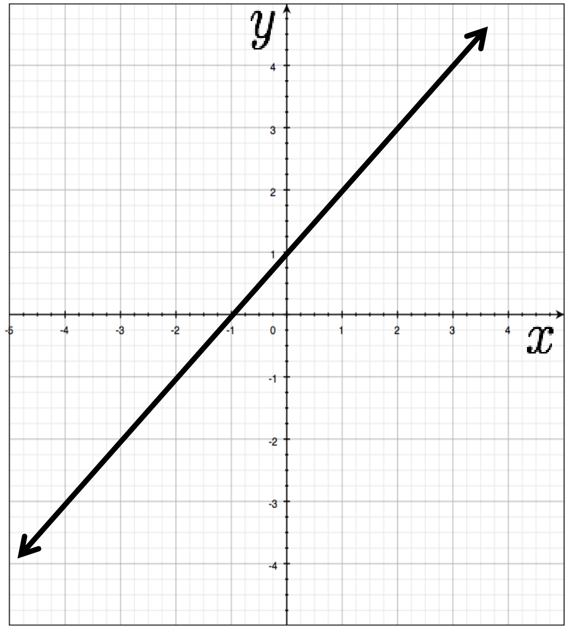
Source : CMU2019

Espace image vs paramètre

variables

$$y = mx + b$$

paramètres



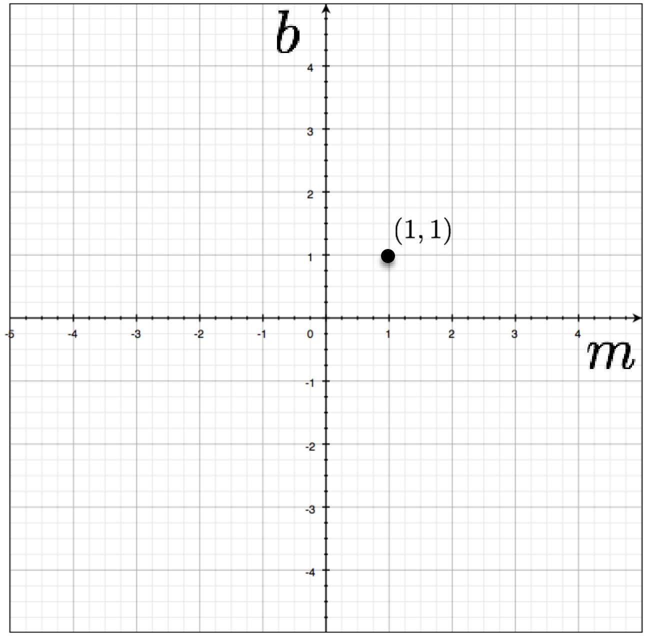
Espace image

Une ligne devient un point

variables

$$y - mx = b$$

paramètres



Espace paramètre

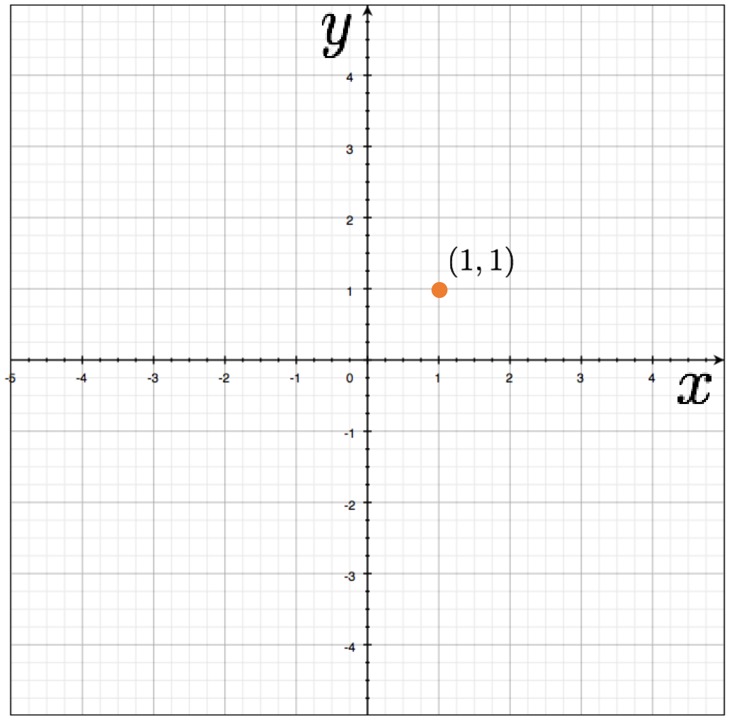
Espaces image et paramètre

- Qu'est-ce que devient un point dans l'espace image lorsque représenté dans l'espace paramètre ?

variables

$$y = mx + b$$

paramètres



Espace image

Espaces image et paramètre (1 point)

variables

$$y = mx + b$$

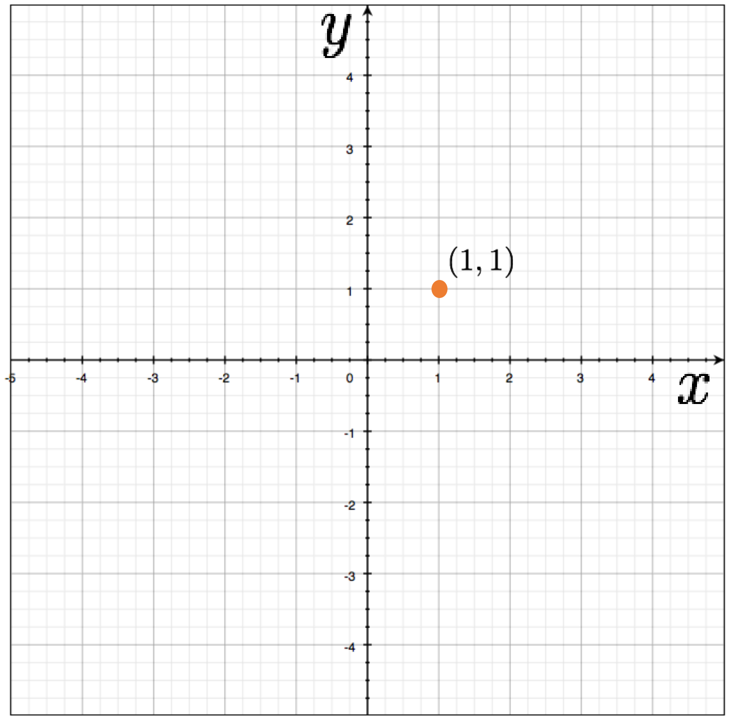
paramètres

variables

$$y - mx = b$$

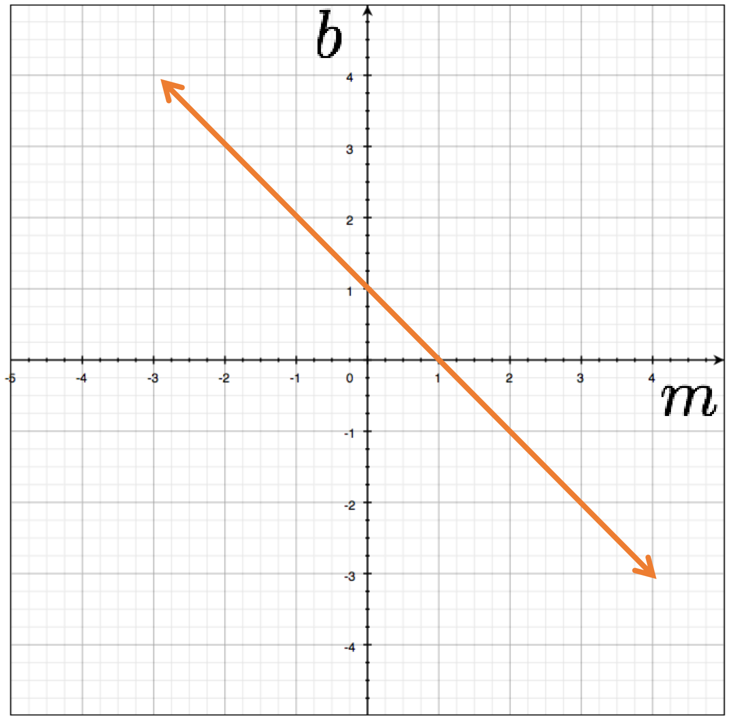
paramètres

Source : CMU2019



Espace image

Un point devient une ligne



Espace paramètre

Espaces image et paramètre (2 points)

variables

$$y = mx + b$$

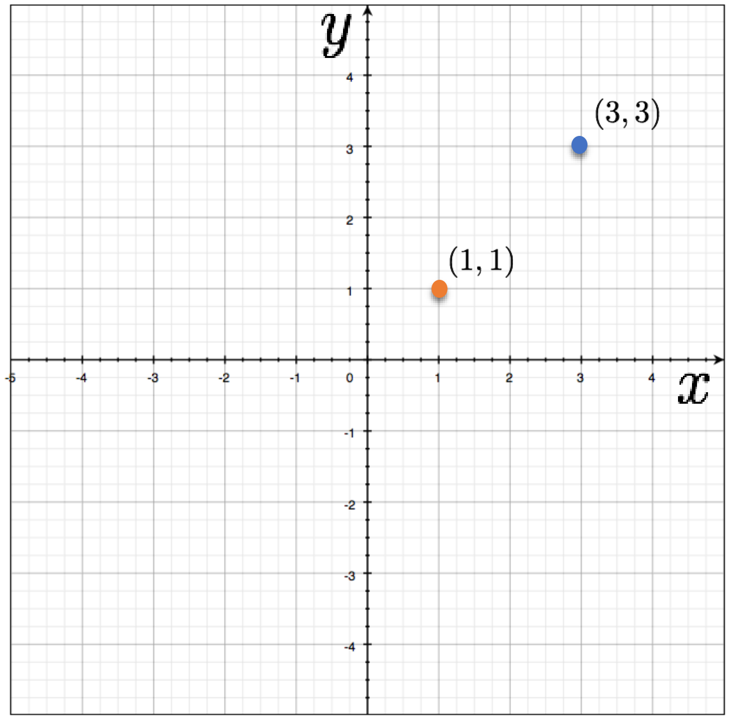
paramètres

variables

$$y - mx = b$$

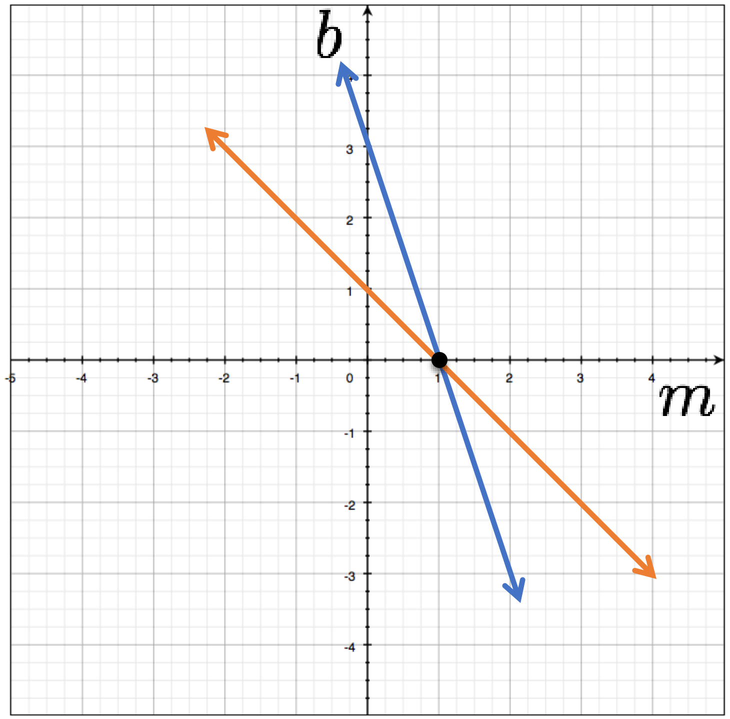
paramètres

Source : CMU2019



Espace image

2 points deviennent ?



Espace paramètre

Espaces image et paramètre (3 points)

variables

$$y = mx + b$$

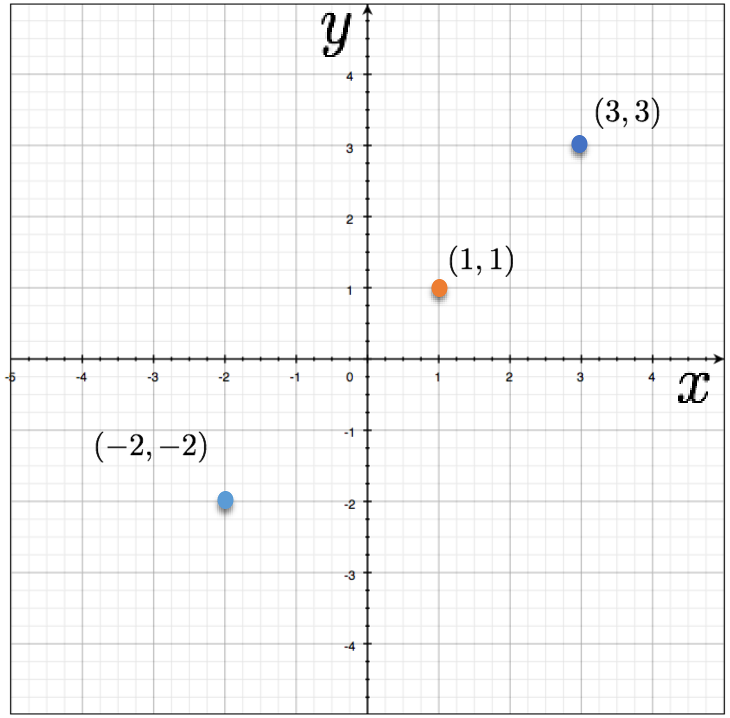
paramètres

variables

$$y - mx = b$$

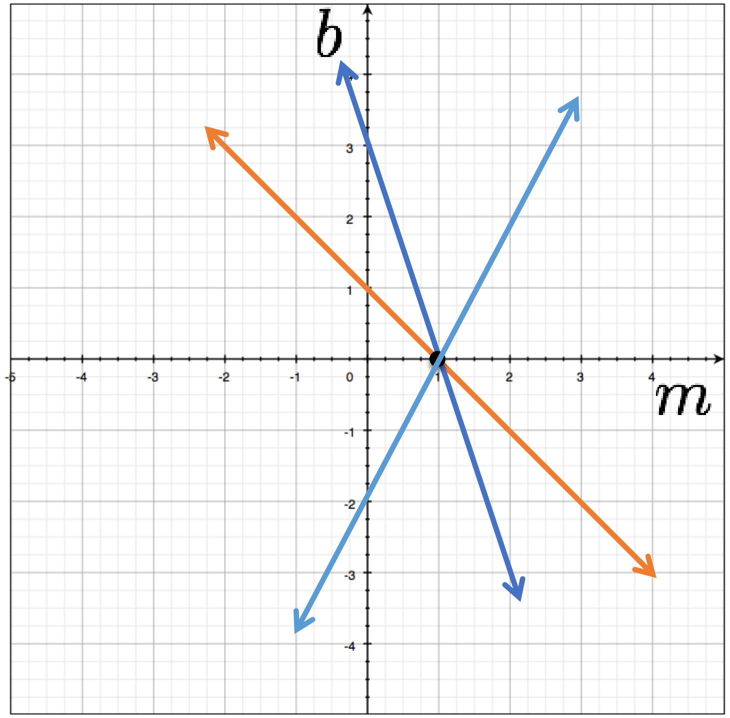
paramètres

Source : CMU2019



Espace image

3 points
deviennent
?



Espace paramètres

Espaces image et paramètre (4 points)

variables

$$y = mx + b$$

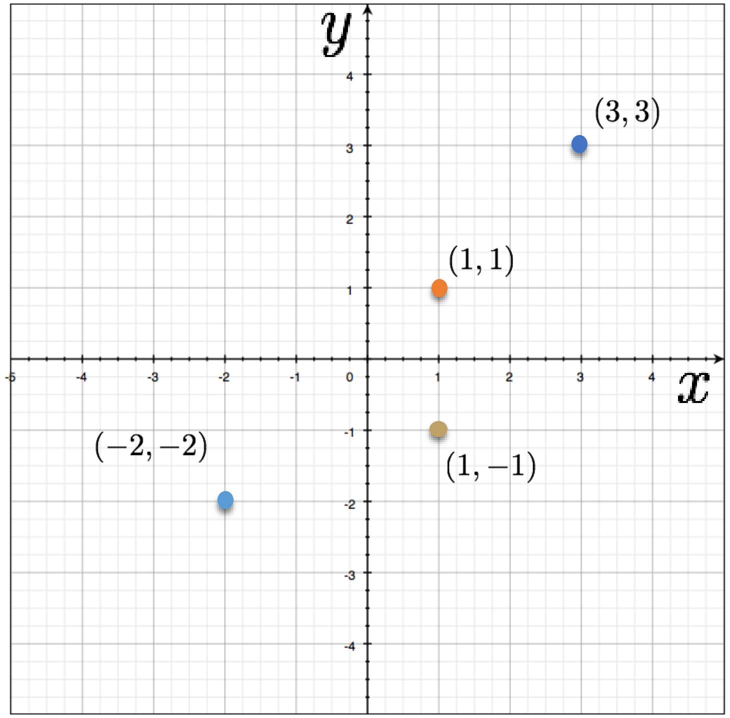
paramètres

variables

$$y - mx = b$$

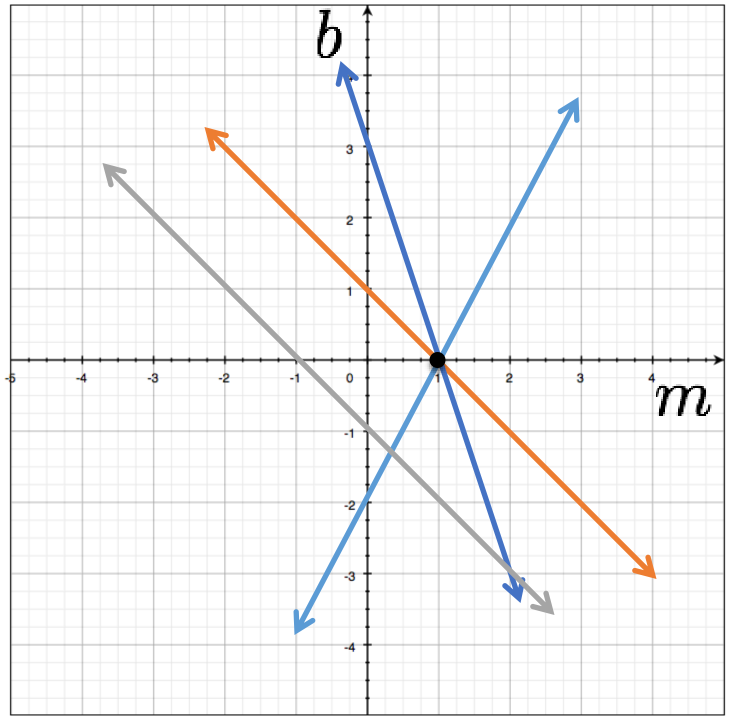
paramètres

Source : CMU2019



Espace image

4 points deviennent ?

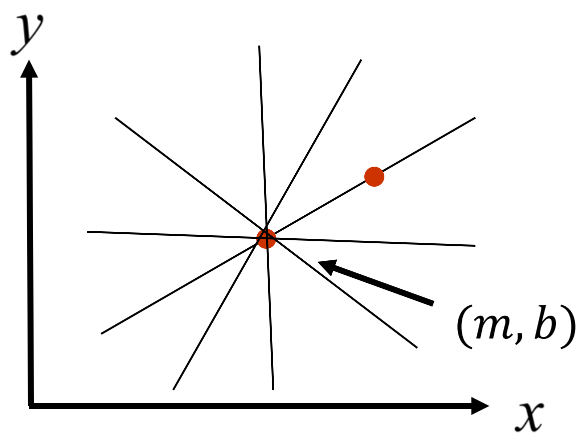


Espace paramètres

Algorithme pour la transformée de Hough

Algorithme (Forme $y = mx + b$)

- Discrétiser l'espace paramètre (m, b)
- Créer une matrice d'accumulation $A(m, b)$
- Initialiser $A(m, b) = 0 \; \forall \; m, b$
- Pour chaque pixel de contours (x_i, y_i)
  - Pour chaque élément de $A(m, b)$
  - si (m, b) sur une ligne: $b = -x_i m + y_i$
    - Ajouter $A(m, b) = A(m, b) + 1$
- Trouver les maximums locaux $A(m, b)$



Espace image

$A(m, b)$

	1						1		
		1				1			
			1		1				
				2					
			1		1				
		1				1			
	1						1		

Source : CMU2019

Problème avec la paramétrisation ?

- Quelle taille doit avoir l’accumulateur pour la paramétrisation $(m$,b) ?

$A(m,c)$

	1						1		
		1				1			
			1		1				
				2					
			1		1				
		1				1			
	1						1		

L’espace de m est gigantesque !
$-\infty \leq m \leq \infty$

L’espace de b est gigantesque !
$-\infty \leq b \leq \infty$

Source : CMU2019

Meilleure paramétrisation : Forme normale

$$x \cos \theta + y \sin \theta = \rho$$

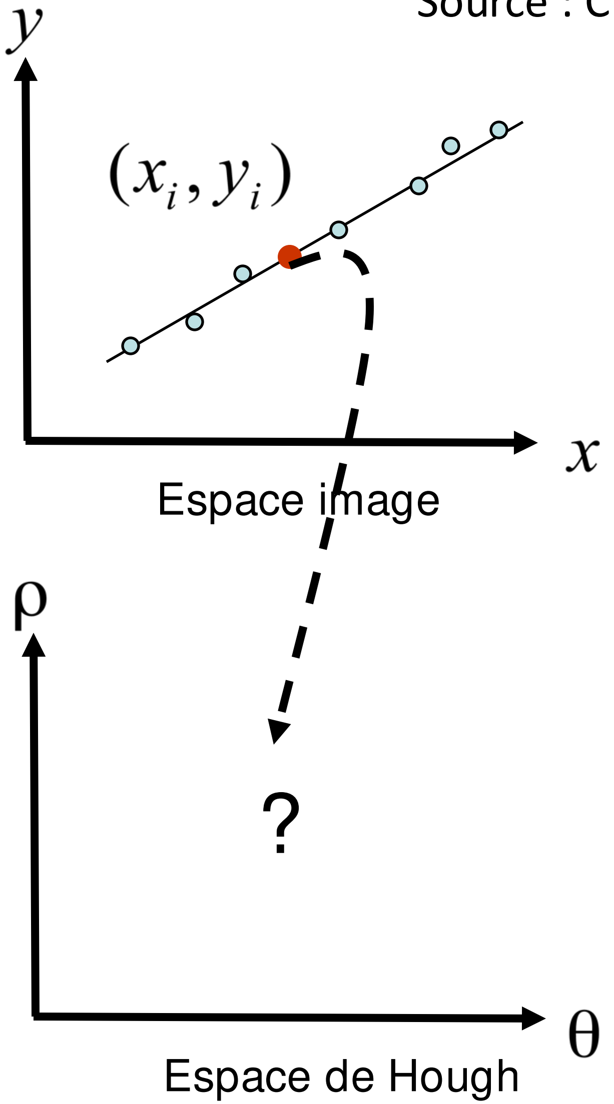
Étant donné un point (x_i, y_i), il faut trouver les paramètres (ρ, θ)

Espace sinusoïdal de Hough

$$0 \leq \theta \leq 2\pi$$
$$0 \leq \rho \leq \rho_{\max}$$

(Matrice accumulatrice de taille finie)

Source : CMU2019



Espaces image et paramètre (1 point)

Source : CMU2019

variables

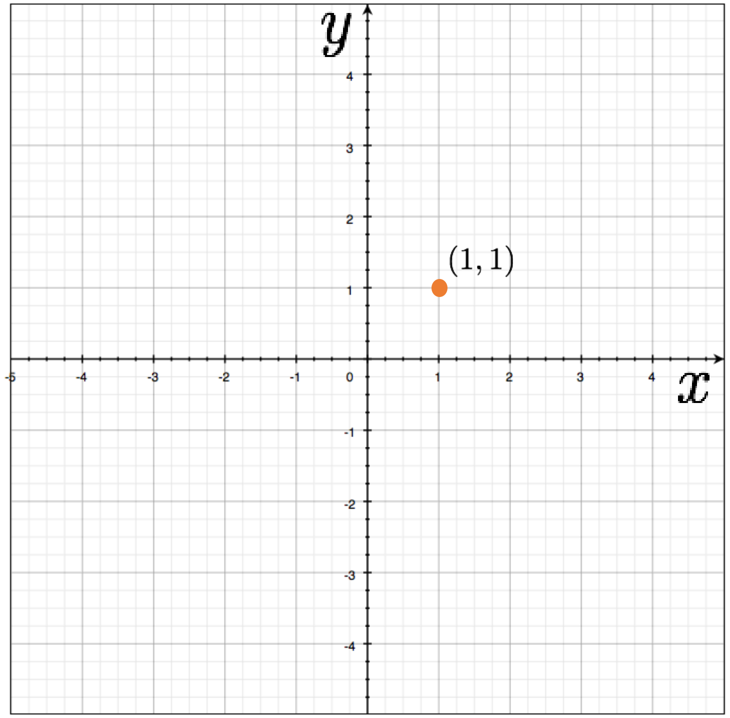
$$y = mx + b$$

paramètres

paramètres

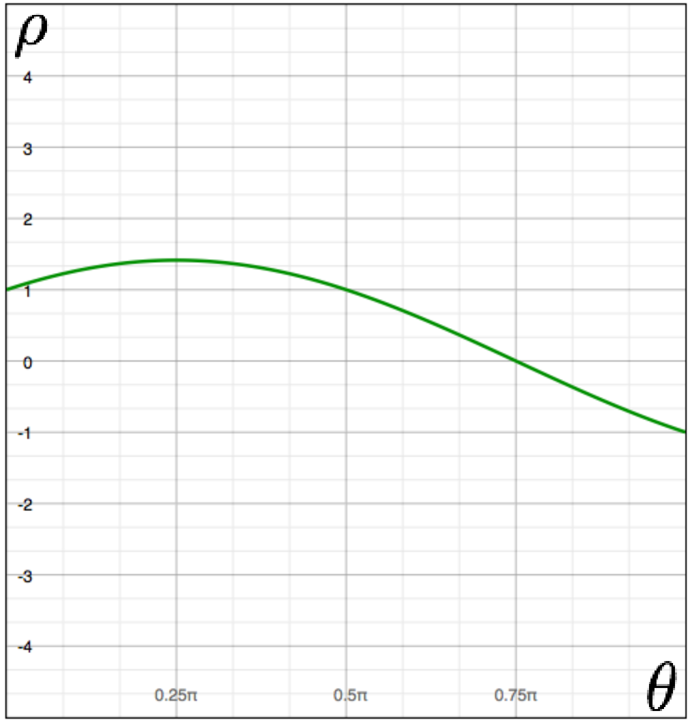
$$x\cos\theta + y\sin\theta = \rho$$

variables



Espace image

Un point devient une onde



Espace paramètre

Espaces image et paramètre (1)

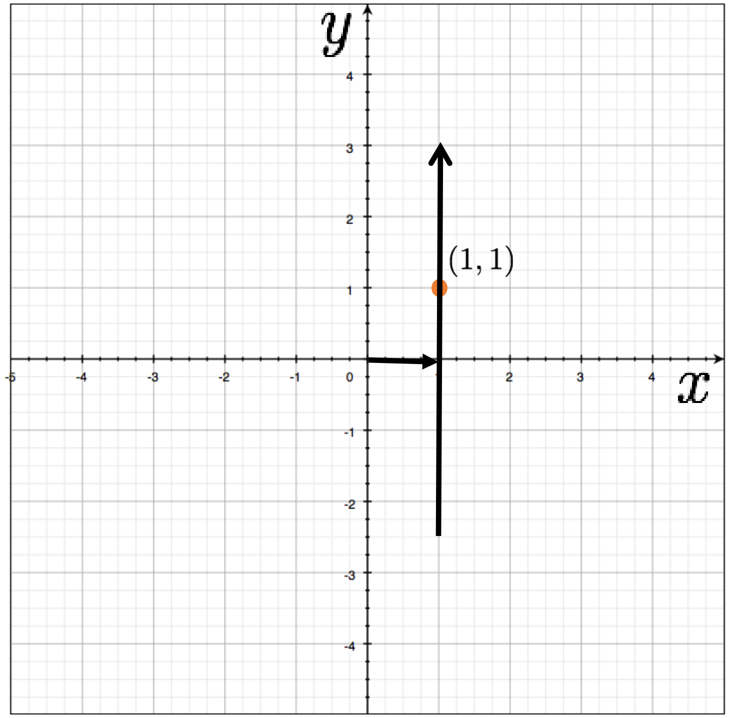
Source : CMU2019

variables

$$y = mx + b$$

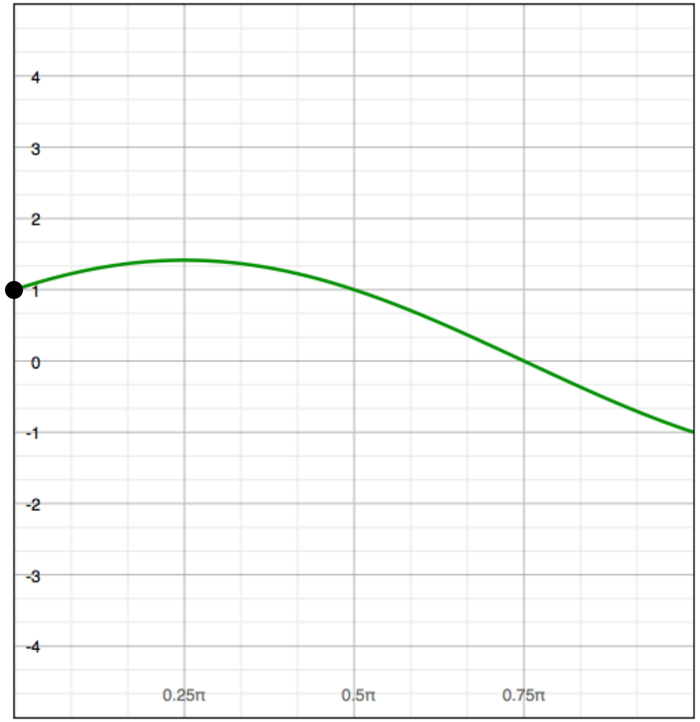
paramètres

$$x \cos \theta + y \sin \theta = \rho$$



Une ligne devient ?

Un point



Espace image

Espace paramètre

Espaces image et paramètre (2)

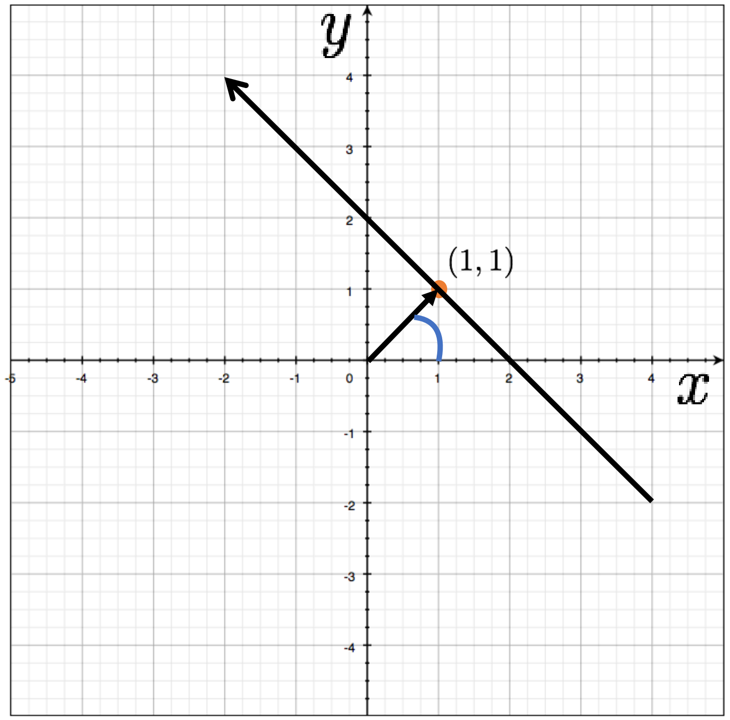
Source : CMU2019

variables

$$y = mx + b$$

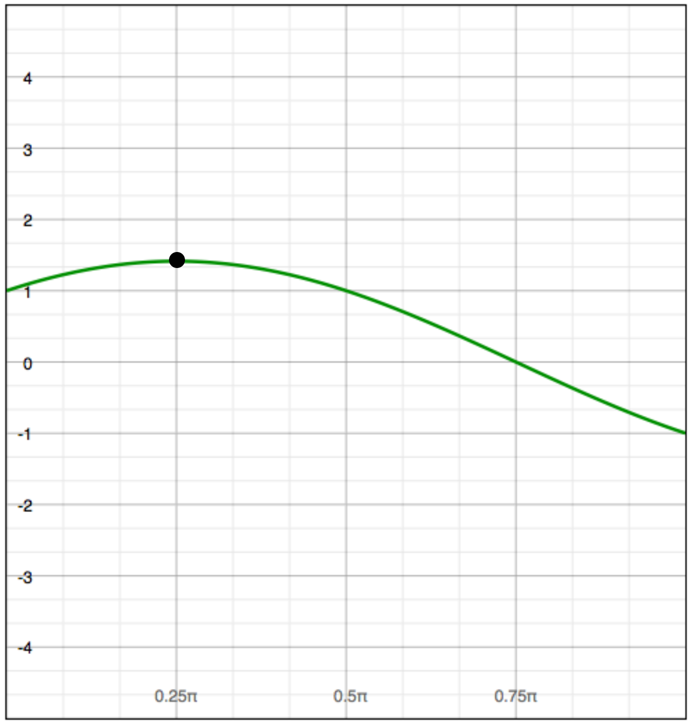
paramètres

$$x \cos\theta + y \sin\theta = \rho$$



Espace image

Une ligne devient un point



Espace paramètre

Espaces image et paramètre (3)

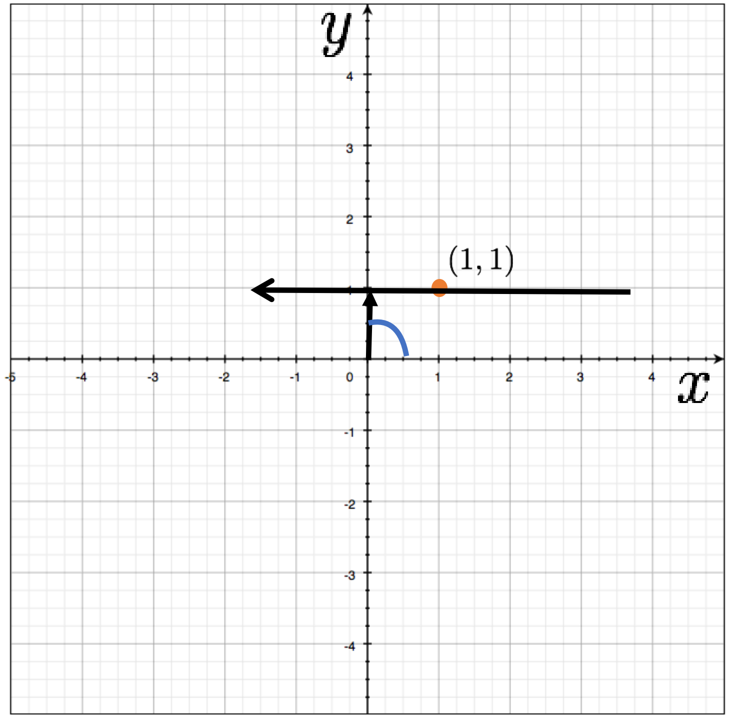
Source : CMU2019

variables

$$y = mx + b$$

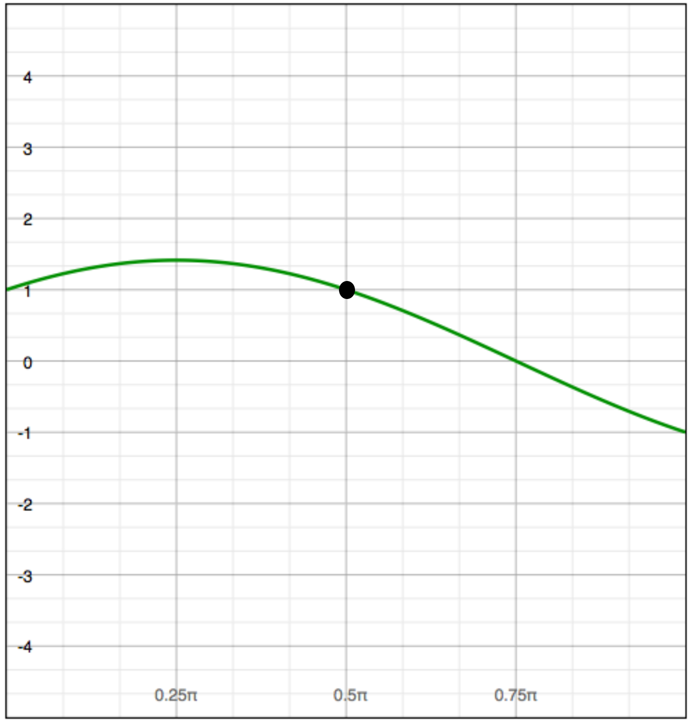
paramètres

$$x \cos\theta + y \sin\theta = \rho$$



Espace image

Une ligne devient un point



Espace paramètre

Espaces image et paramètre (4)

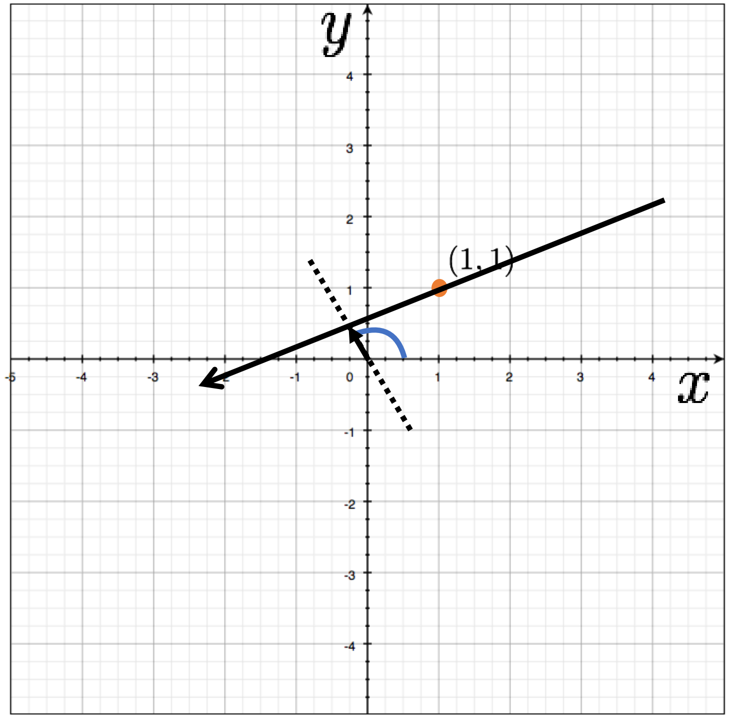
Source : CMU2019

variables

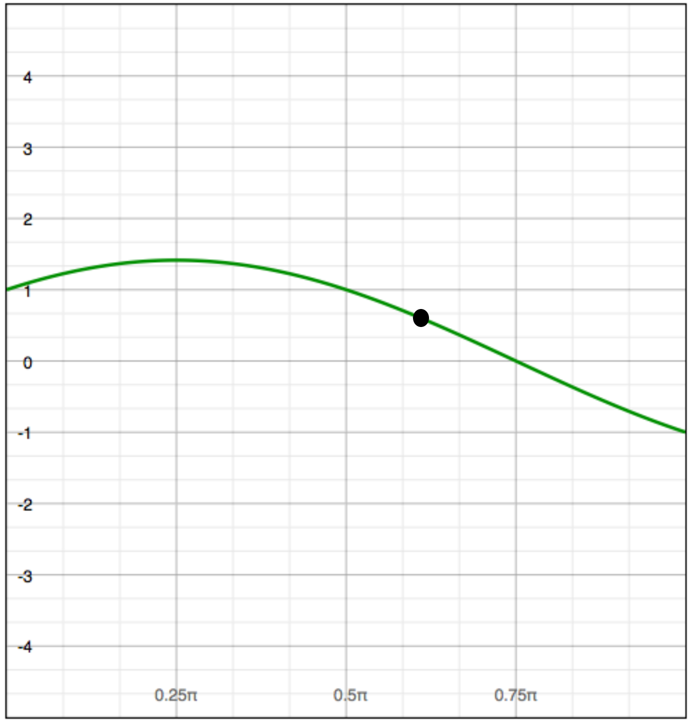
$$y = mx + b$$

paramètres

$$x\cos\theta + y\sin\theta = \rho$$



Une ligne devient un point



Espace image

Espace paramètre

Espaces image et paramètre (5)

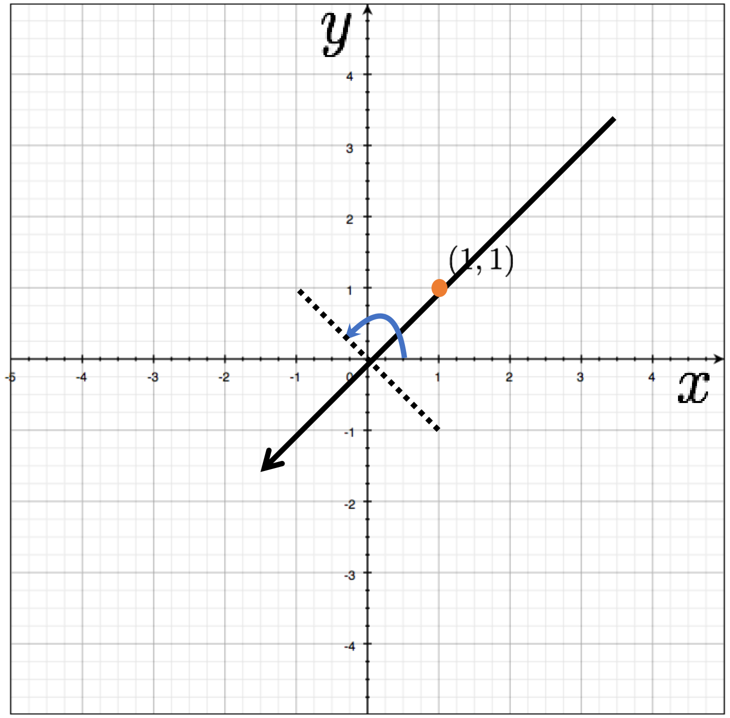
Source : CMU2019

variables

$$y = mx + b$$

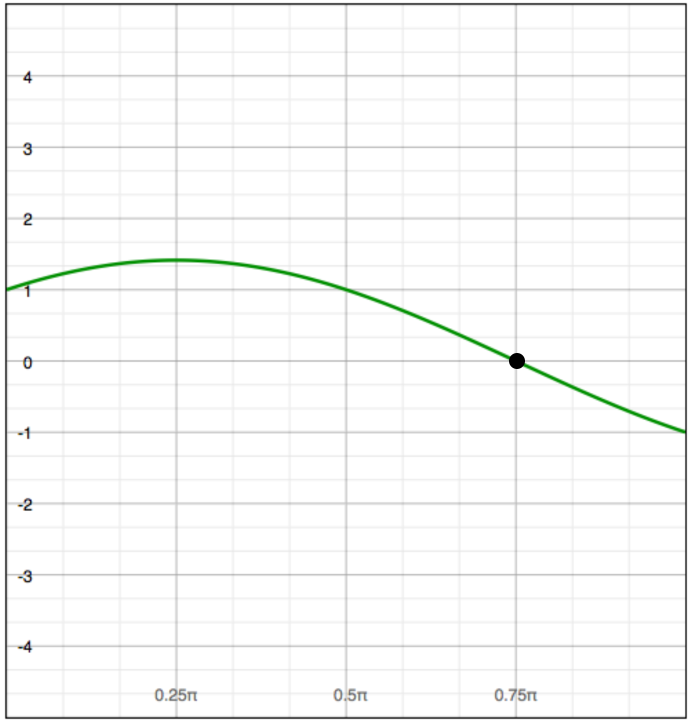
paramètres

$$x \cos\theta + y \sin\theta = \rho$$



Espace image

Une ligne devient un point



Espace paramètre

Espaces image et paramètre (6)

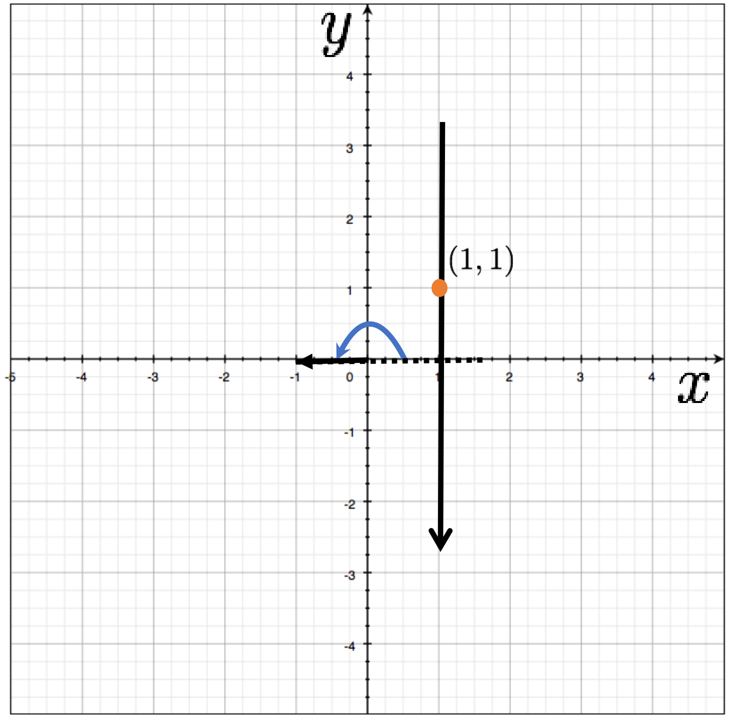
Source : CMU2019

variables

$$y = mx + b$$

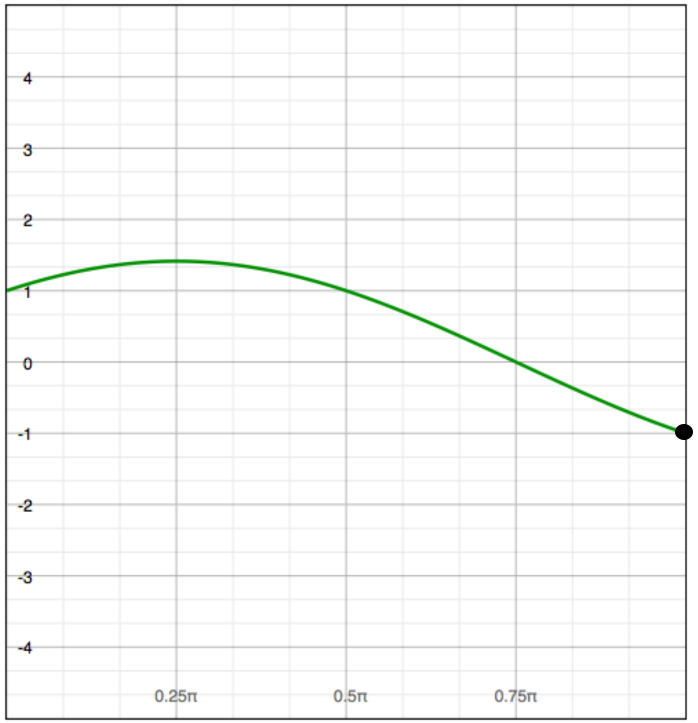
paramètres

$$x \cos\theta + y \sin\theta = \rho$$



Espace image

Une ligne devient un point



Espace paramètre

Espaces image et paramètre

Source : CMU2019

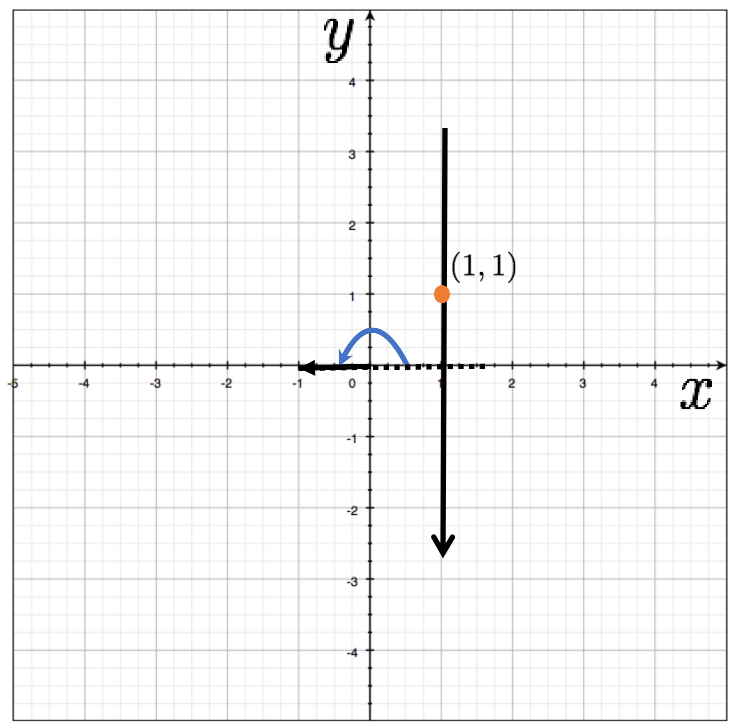
variables

$$y = mx + b$$

paramètres

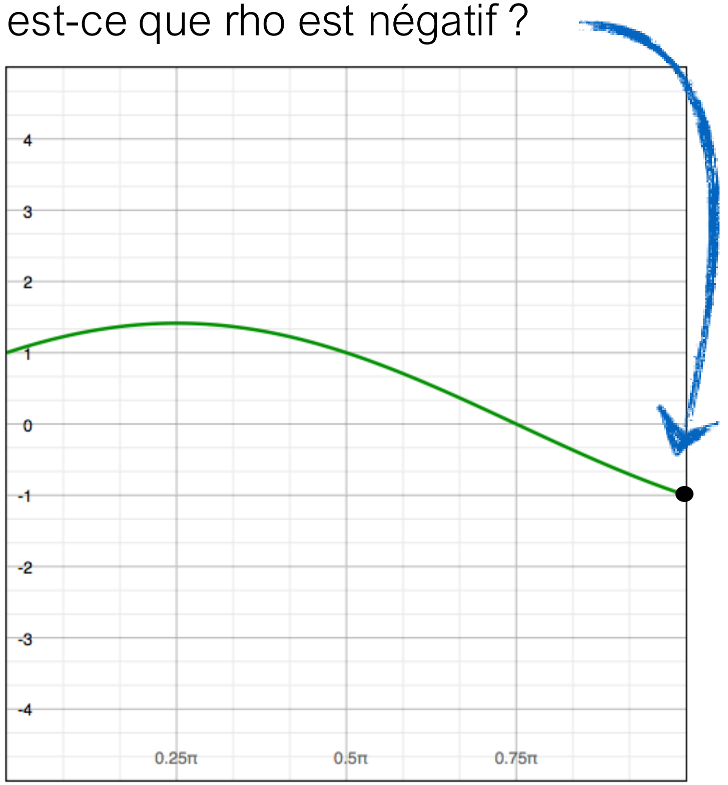
$$x \cos\theta + y \sin\theta = \rho$$

Pourquoi est-ce que rho est négatif ?



Espace image

Une ligne devient un point



Espace paramètre

Espaces image et paramètre

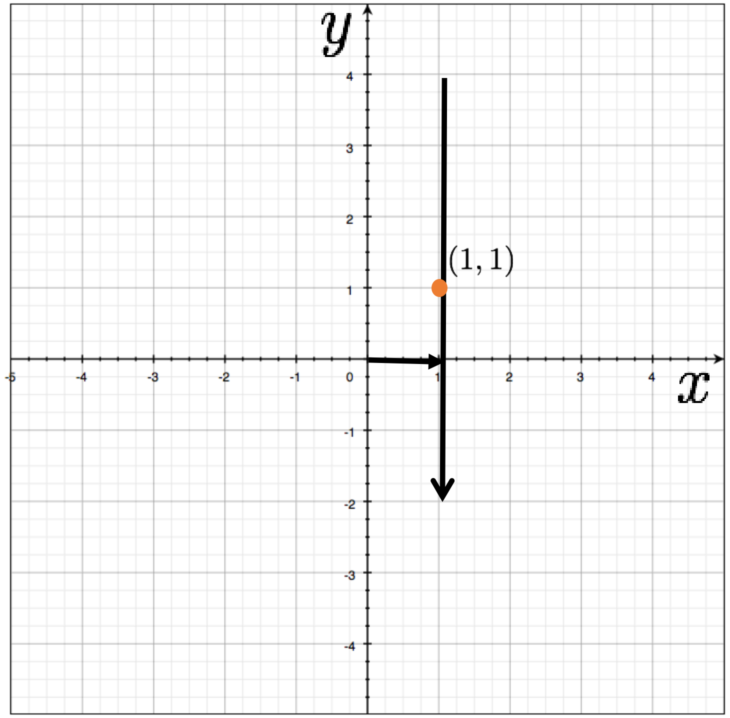
Source : CMU2019

variables

$$y = mx + b$$

paramètres

$$x \cos\theta + y \sin\theta = \rho$$



Une ligne devient un point

Espace image

Espace paramètre

Rayons négatifs

- Il y a 2 façons de décrire la même ligne
- Version positive de ρ

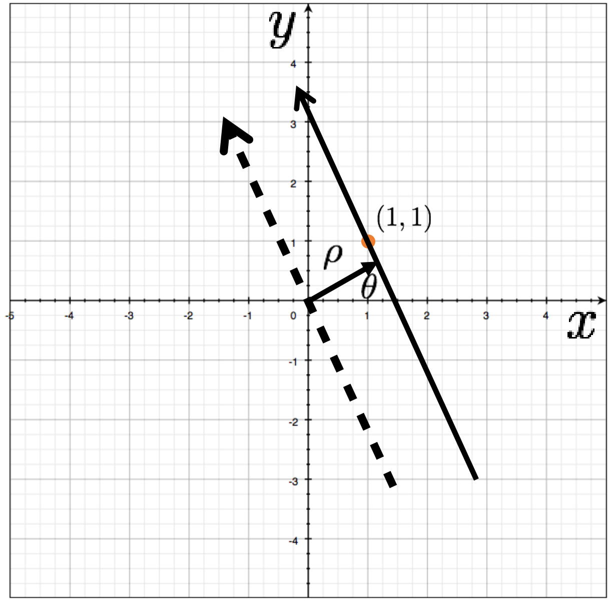
$$x\cos\theta + y\sin\theta = \rho$$

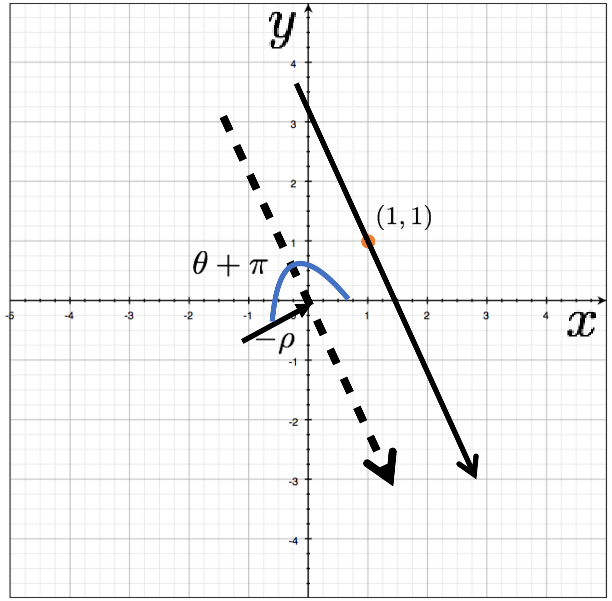
- Version négative de ρ

$$x\cos(\theta+\pi) + y\sin(\theta+\pi) = -\rho$$

Rappel

$$\sin(\theta) = -\sin(\theta+\pi)$$
$$\cos(\theta) = -\cos(\theta+\pi)$$





Source : CMU2019

Espaces image et paramètre (1 point)

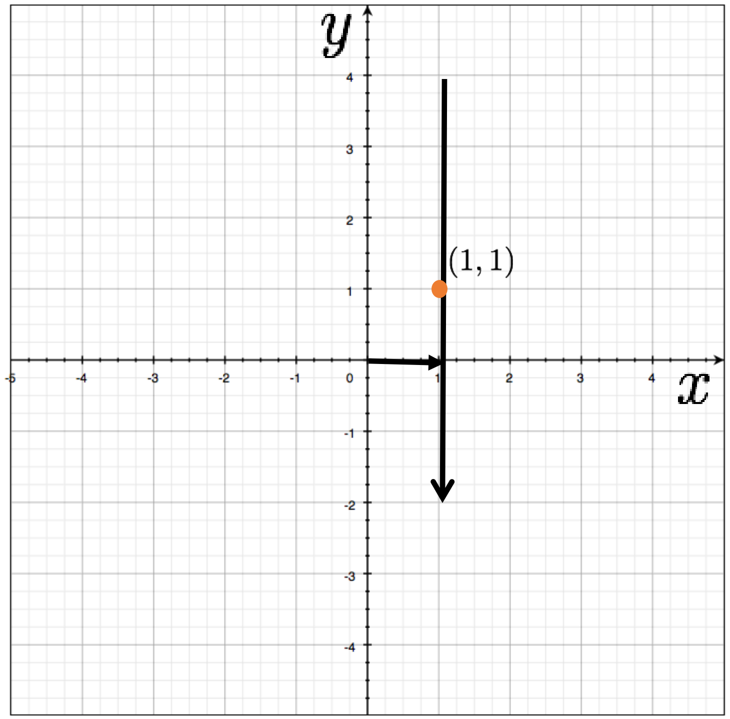
Source : CMU2019

variables

$$y = mx + b$$

paramètres

$$x\cos\theta + y\sin\theta = \rho$$



Espace image

Une ligne devient un point

Espace paramètre

Espaces image et paramètre (2 points)

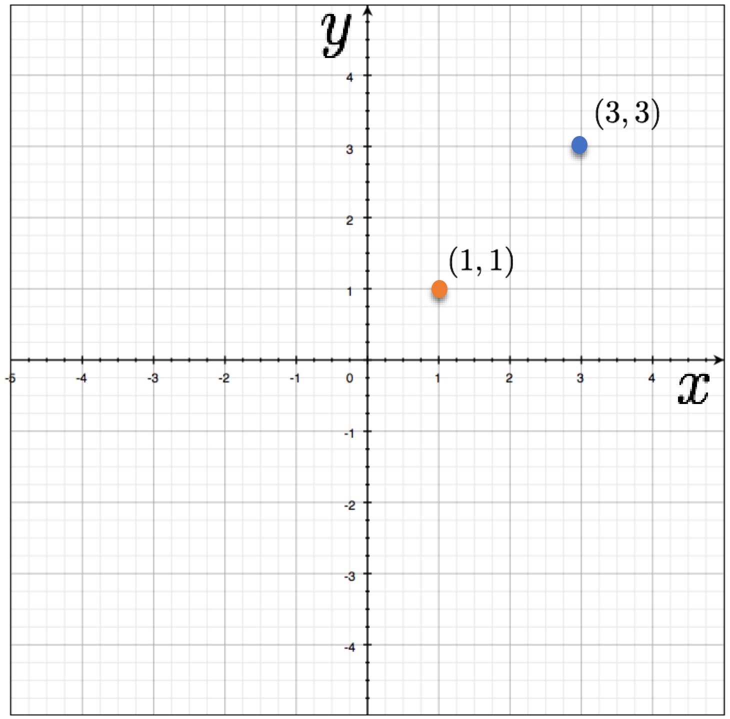
$$\underbrace{y}_{\text{variables}} = m\underbrace{x}_{\text{variables}} + b$$

variables

$$y = mx + b$$

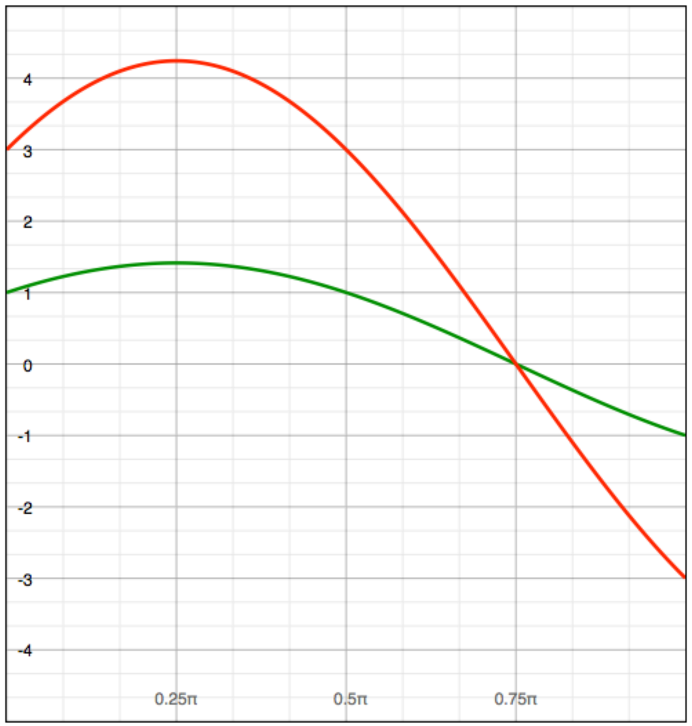
Paramètres

Source : CMU2019



Espace image

2 points deviennent ?



Espace paramètre

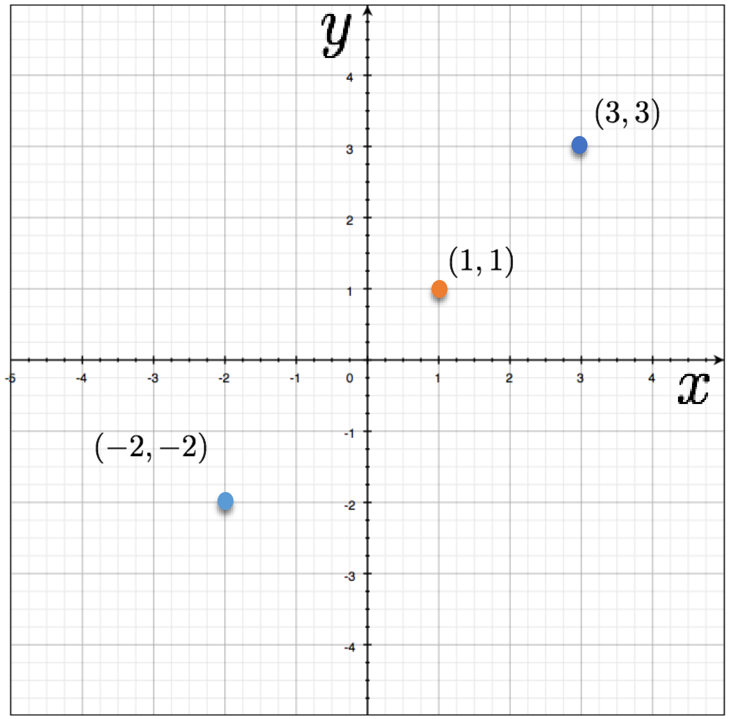
Espaces image et paramètre (3 points)

variables

$$y = mx + b$$

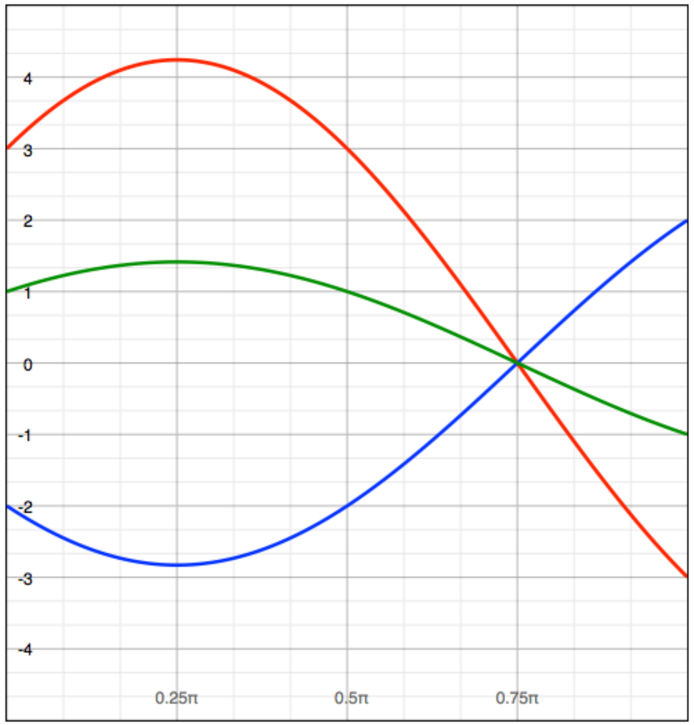
paramètres

Source : CMU2019



Espace image

3 points ?



Espace paramètre

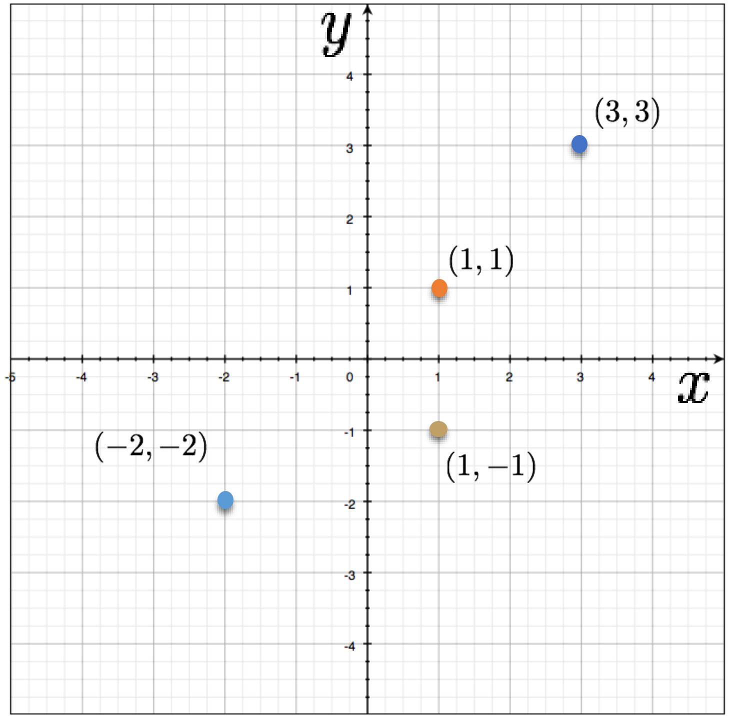
Espaces image et paramètre (4 points)

variables

$$y = mx + b$$

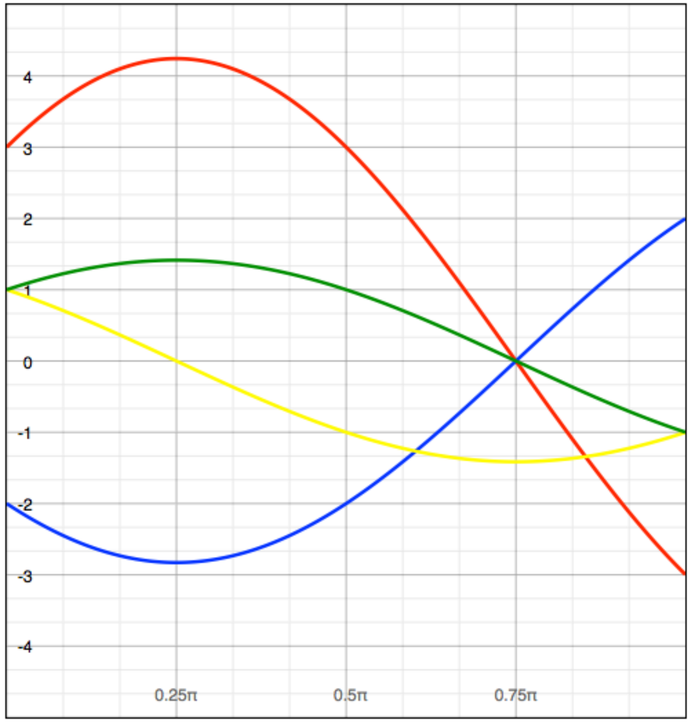
paramètres

Source : CMU2019



Espace image

4 points ?



Espace paramètre

Implémentation

1. Initialiser l'accumulateur $H(\rho, \theta)$ à zéro
2. Pour chaque pixel de contour (x, y) de l'image

   Pour $\theta = 0$ à 180

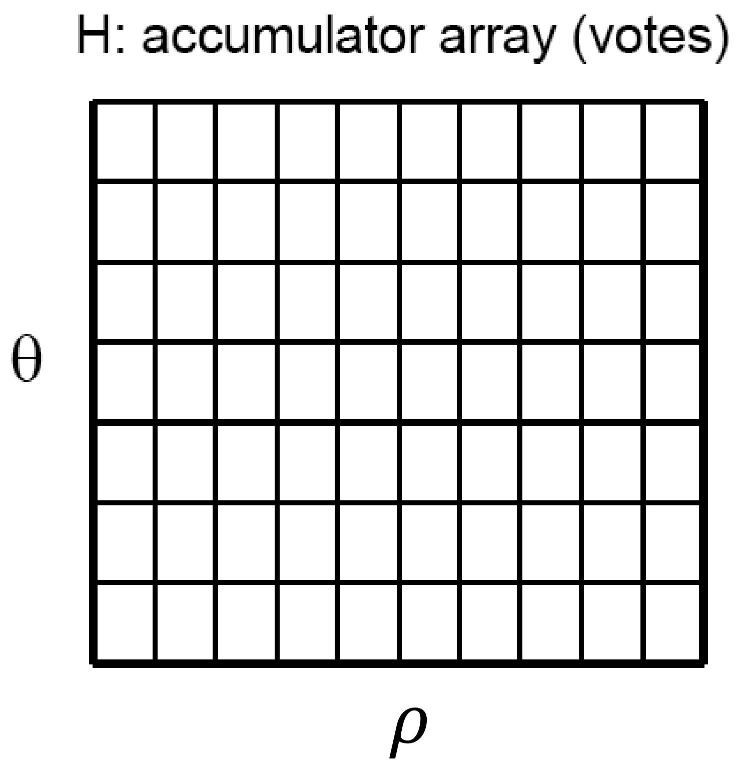
   $\rho = x \cos\theta + y \sin\theta$

   $H(\rho, \theta) = H(\rho, \theta) + 1$

   fin

   fin
3. Trouver les valeurs de (ρ, θ) où $H(\rho, \theta)$ est un maximum local
4. La ligne détectée dans l'image est donnée par :

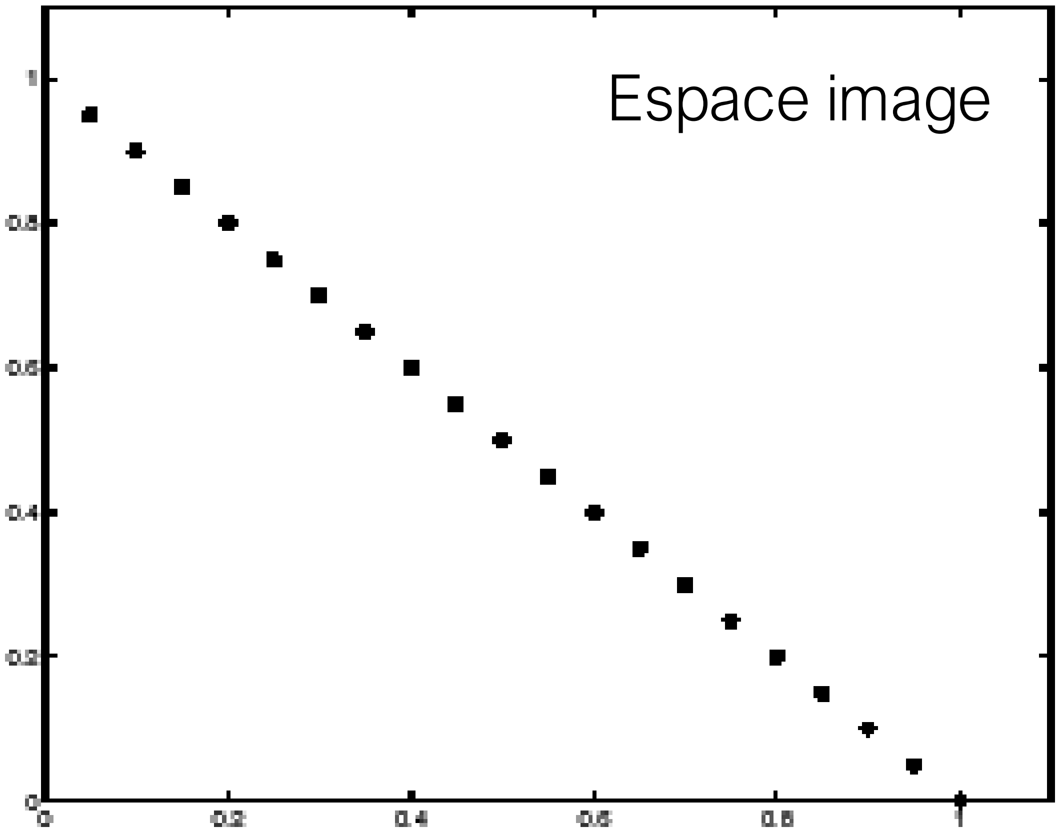
   $\rho = x \cos\theta + y \sin\theta$

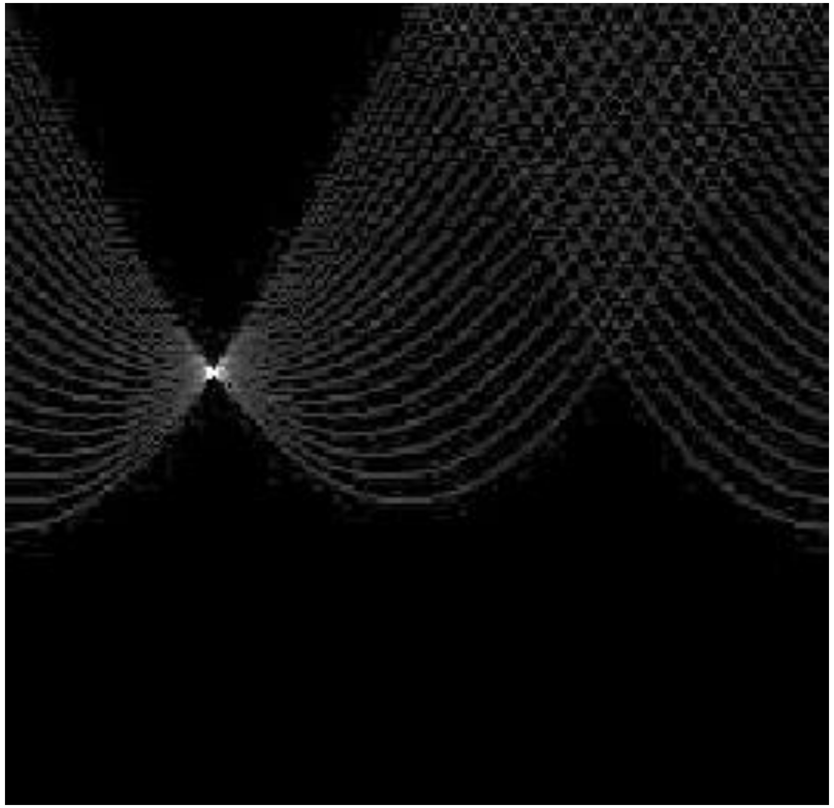


Note : Attention aux coordonnées. L'origine de l'image est en haut à gauche.

Python et transformée de Hough

- `skimage.transform.hough_line` (Documentation)



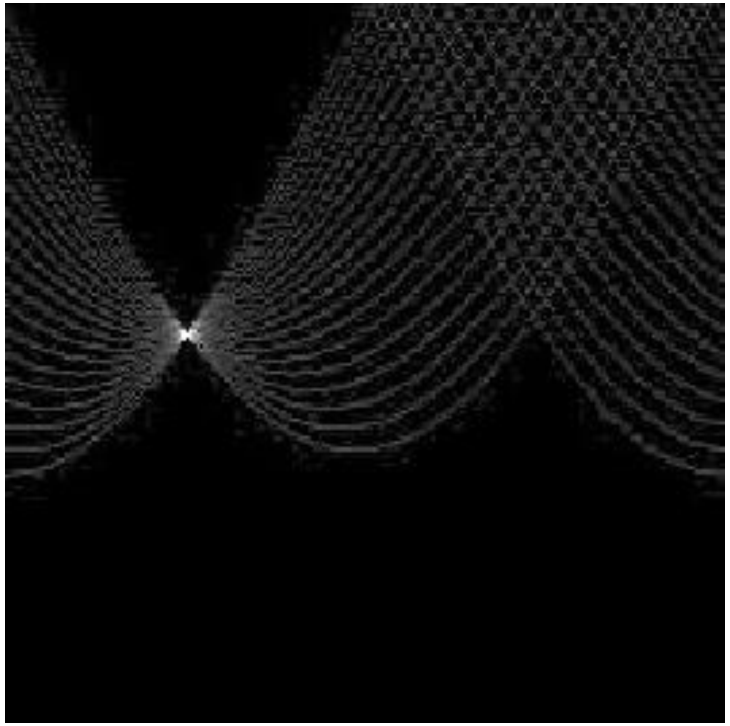


Votes

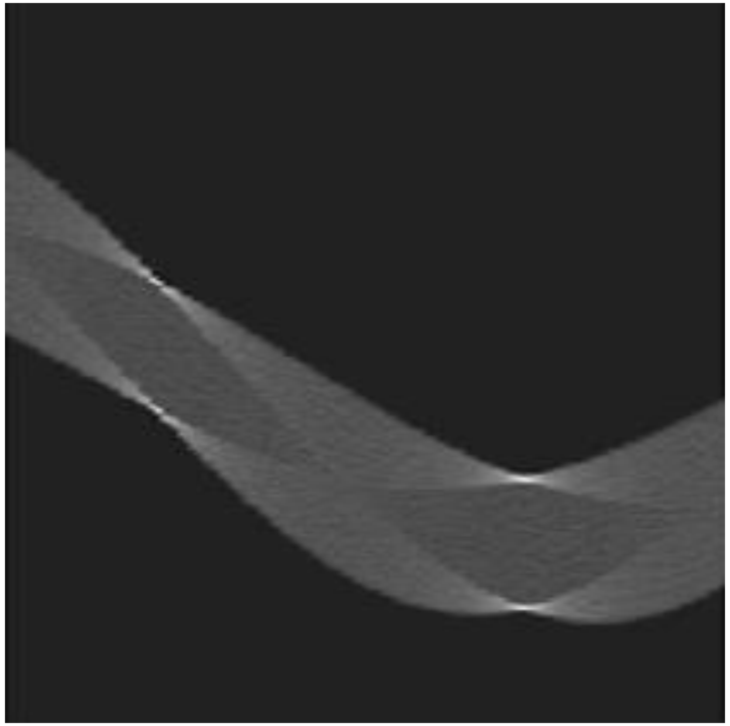
Formes de base (dans l'espace paramètre)

- Quels sont les objets représentés ?

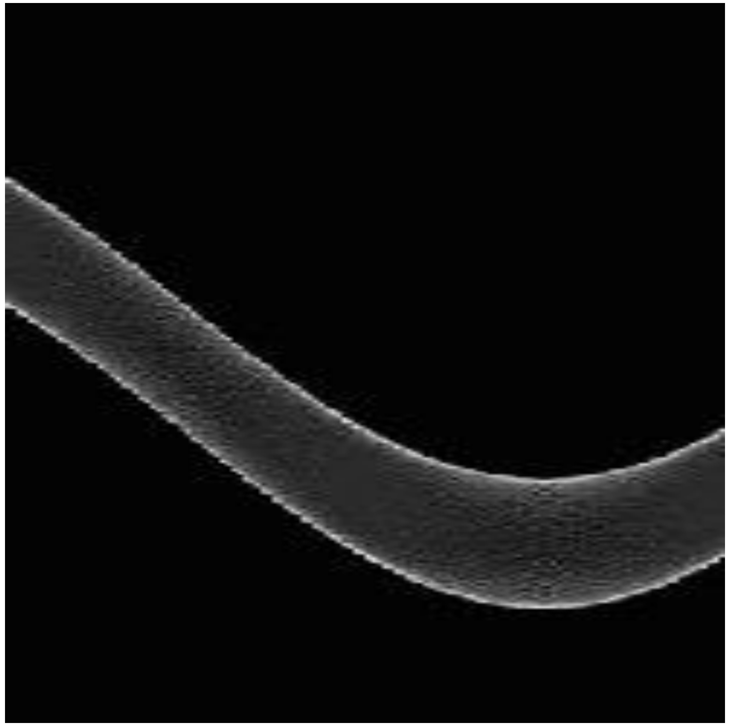
Source : CMU2019



Ligne

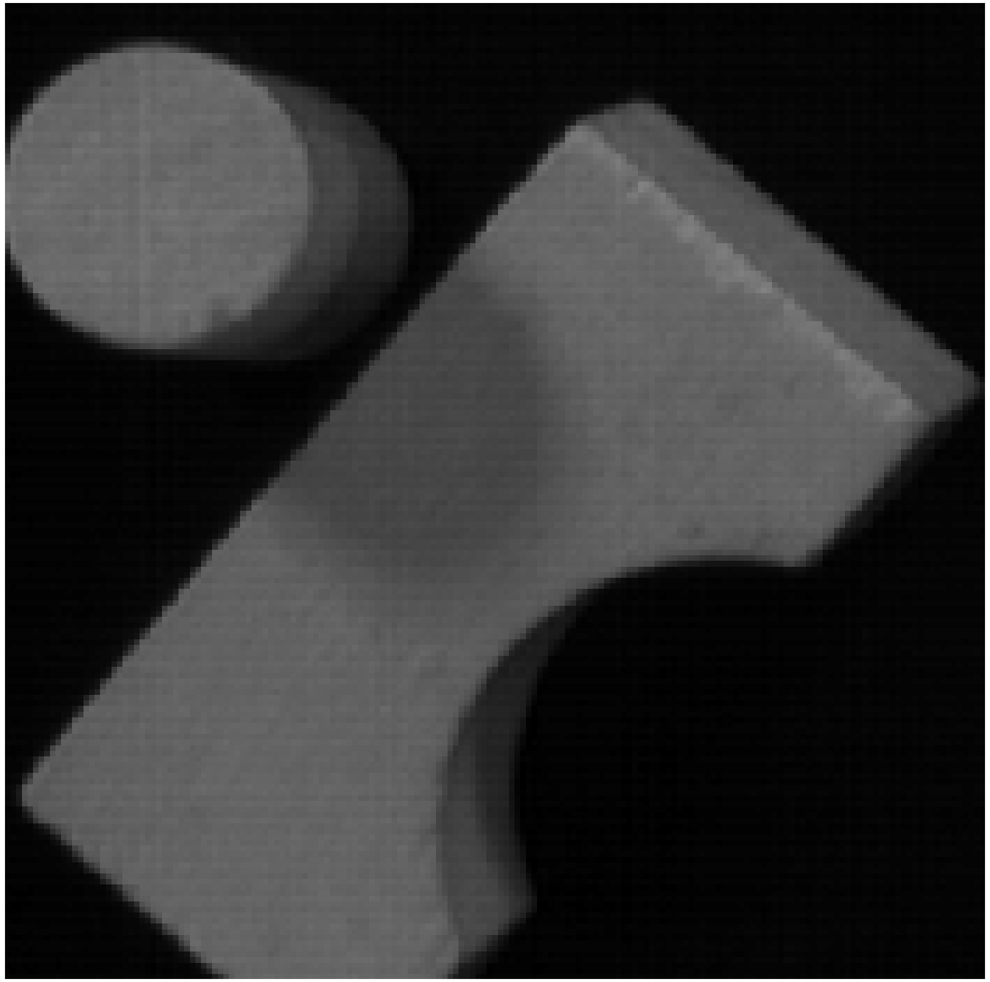


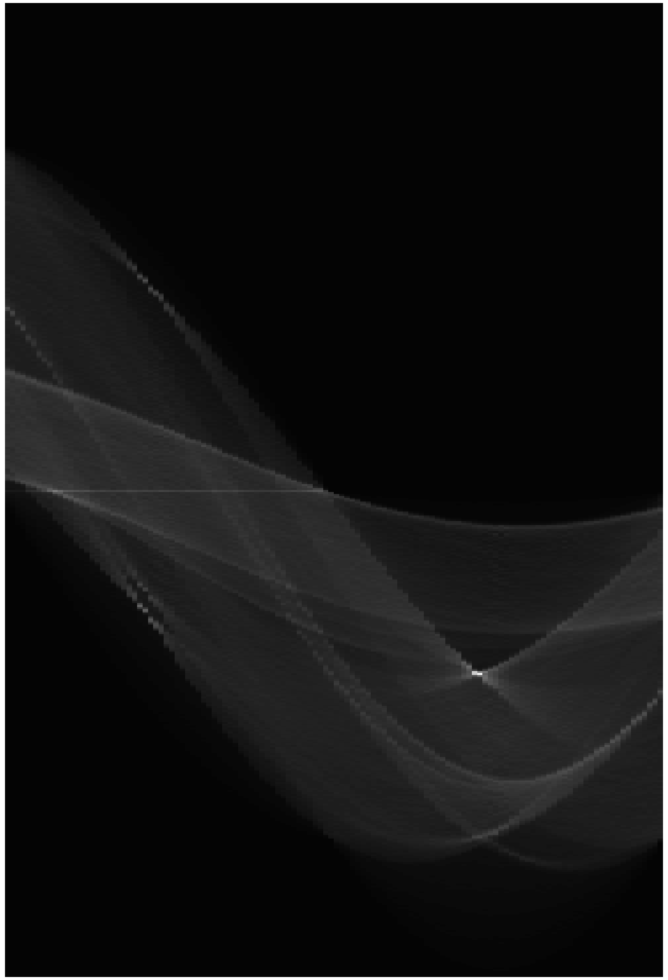
Rectangle



Cercle

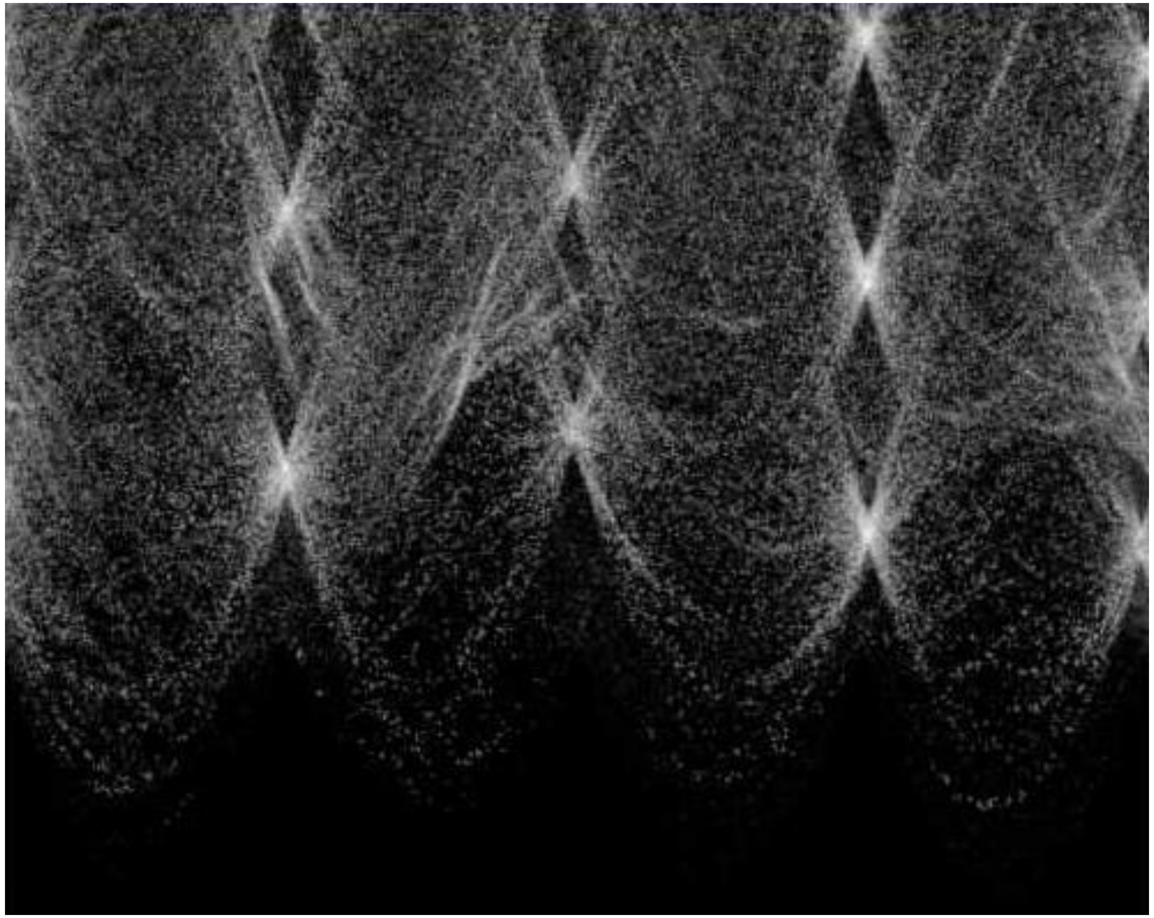
Transformée de Hough d'une ligne
Formes de base





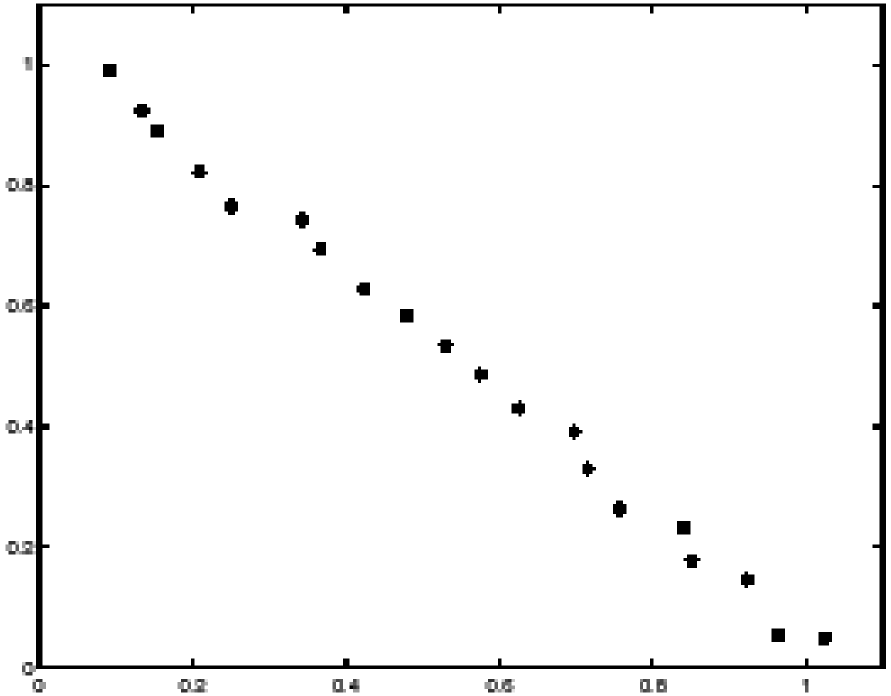
Source : CMU2019

Image plus complexe

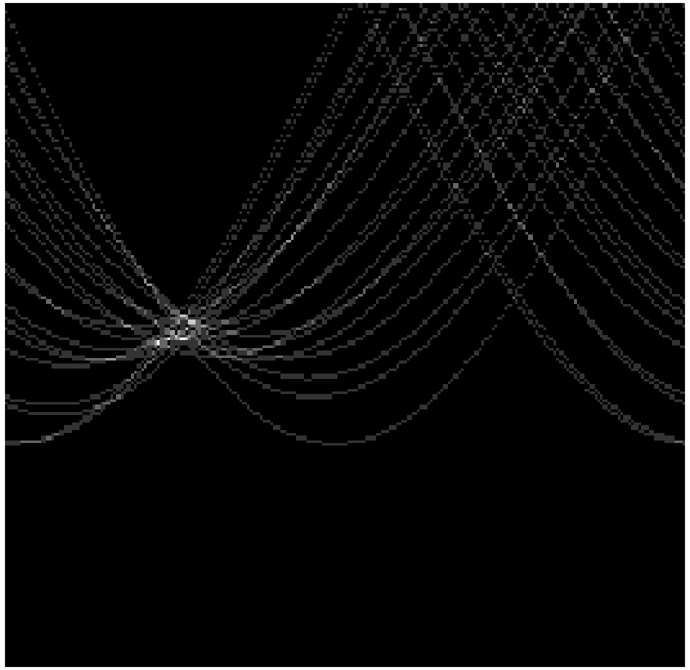


Source : CMU2019

En pratique, les mesures sont bruitées



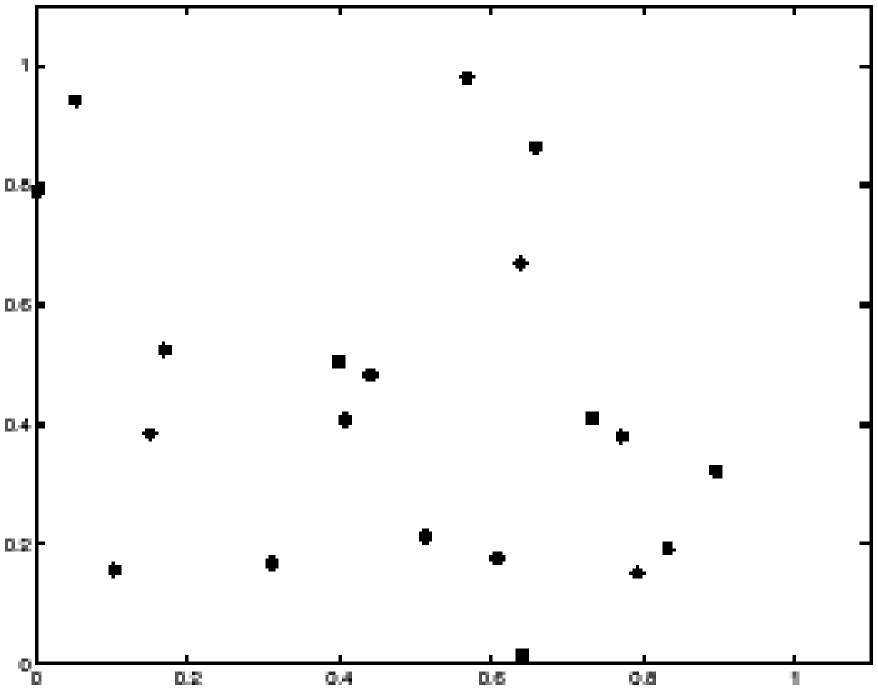
Espace image



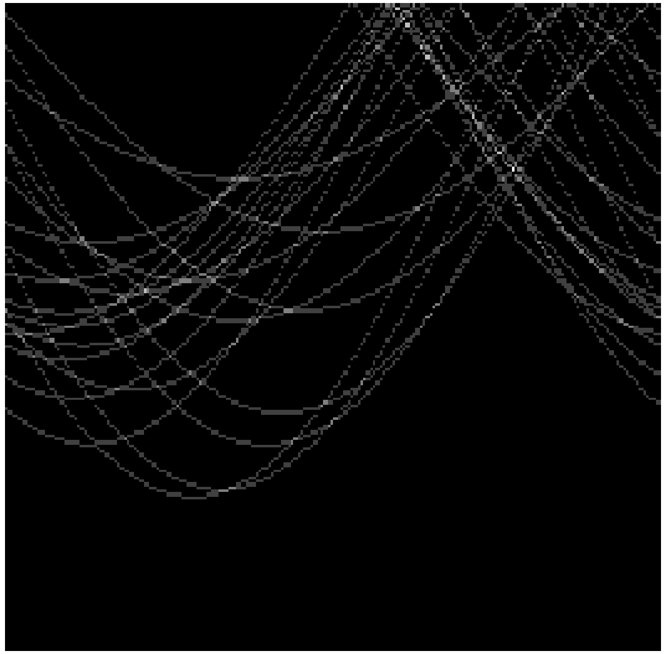
Votes

Ici, il y a trop de bruit…

- Plus le bruit augmente, et moins il y aura de votes pour la droite à extraire.



Espace image



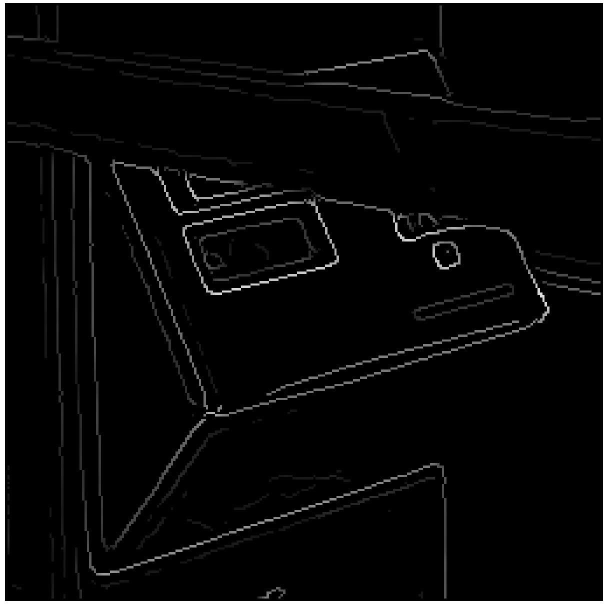
Votes

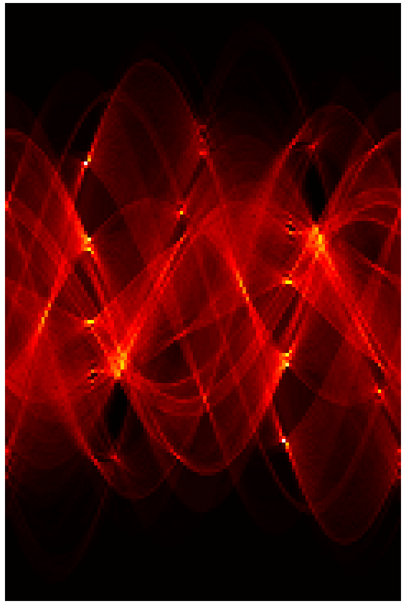
Source : CMU2019

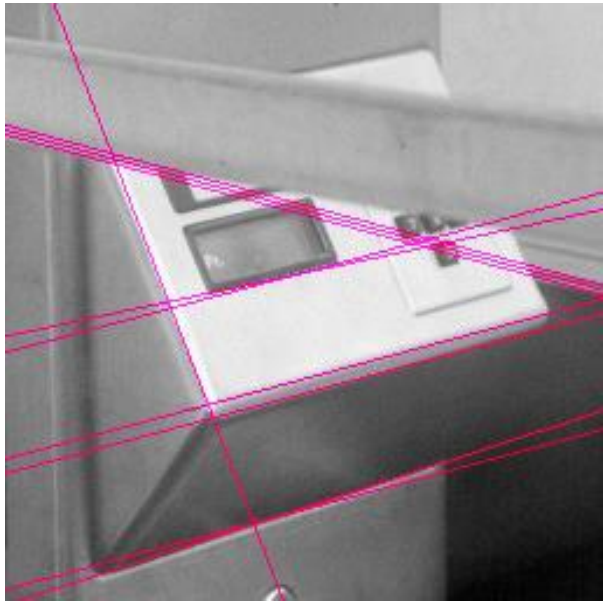
Source : CMU2019

Exemple d'application de la transformée de Hough : Détection d'un objet rectangulaire


Original

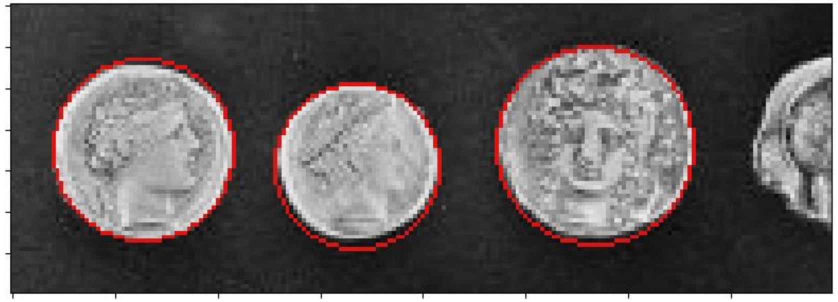

Contours


Espace paramètre

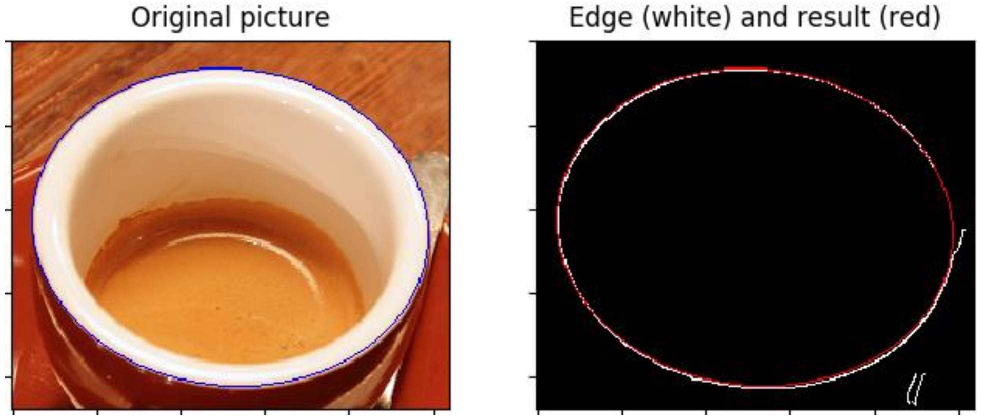

Lignes de Hough

Autres types de transformées de Hough

- Cercle (**skimage.transform.hough_circle**)



- Ellipse (**skimage.transform.hough_ellipse**)
- Plan et surface (3D)



Limitation de la transformée de Hough: Très lourd au point de vue computationnel