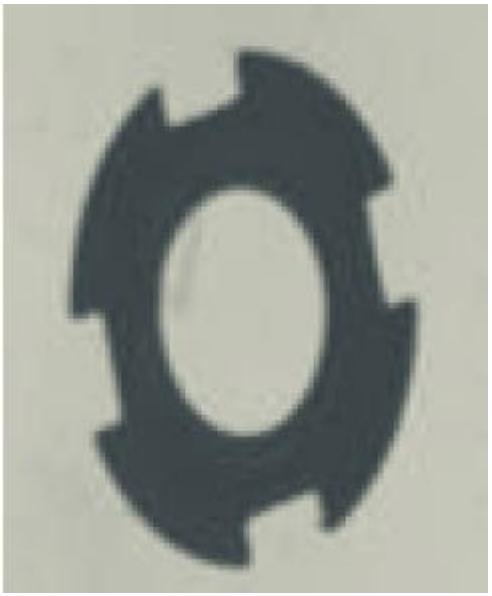
Chapitre 03
Extraction de caractéristiques dans les images

Dr. Aissa Boulmerka
a.boulmerka@centre-univ-mila.dz
2021-2022

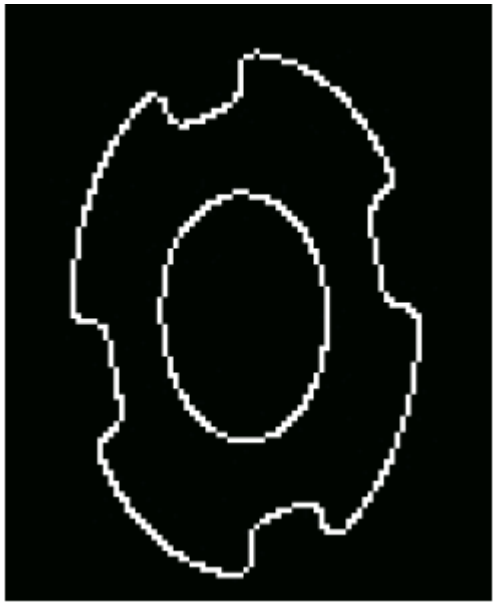
1- EXTRACTION DE CONTOURS

Introduction

$\boldsymbol{f(x,y)}$



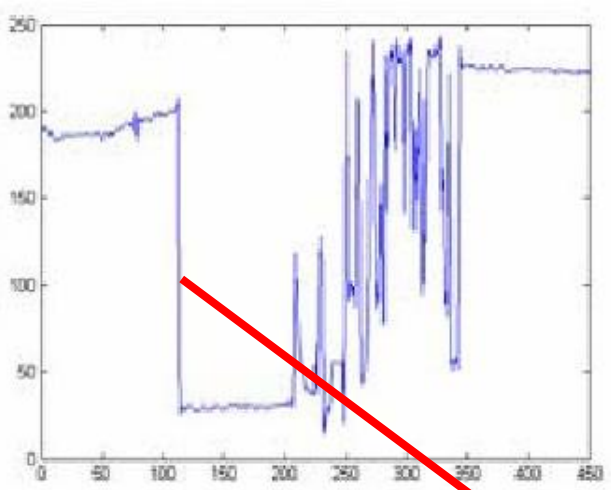
$\boldsymbol{c(x,y)}$



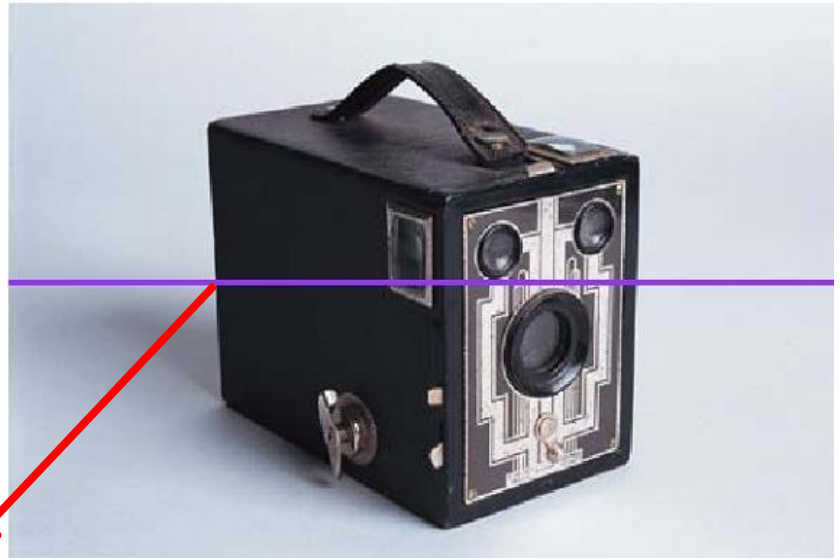
- $\boldsymbol{f(x,y)}$: une image d'entrée
- $\boldsymbol{c(x,y)}$: une image binaire, $c(i,j) \in$ {contour, non-contour}
- On appelle parfois **edge map** l'image $c(x,y)$ étiquettes.

Définition

- Un contour est une variation brusque d'intensité

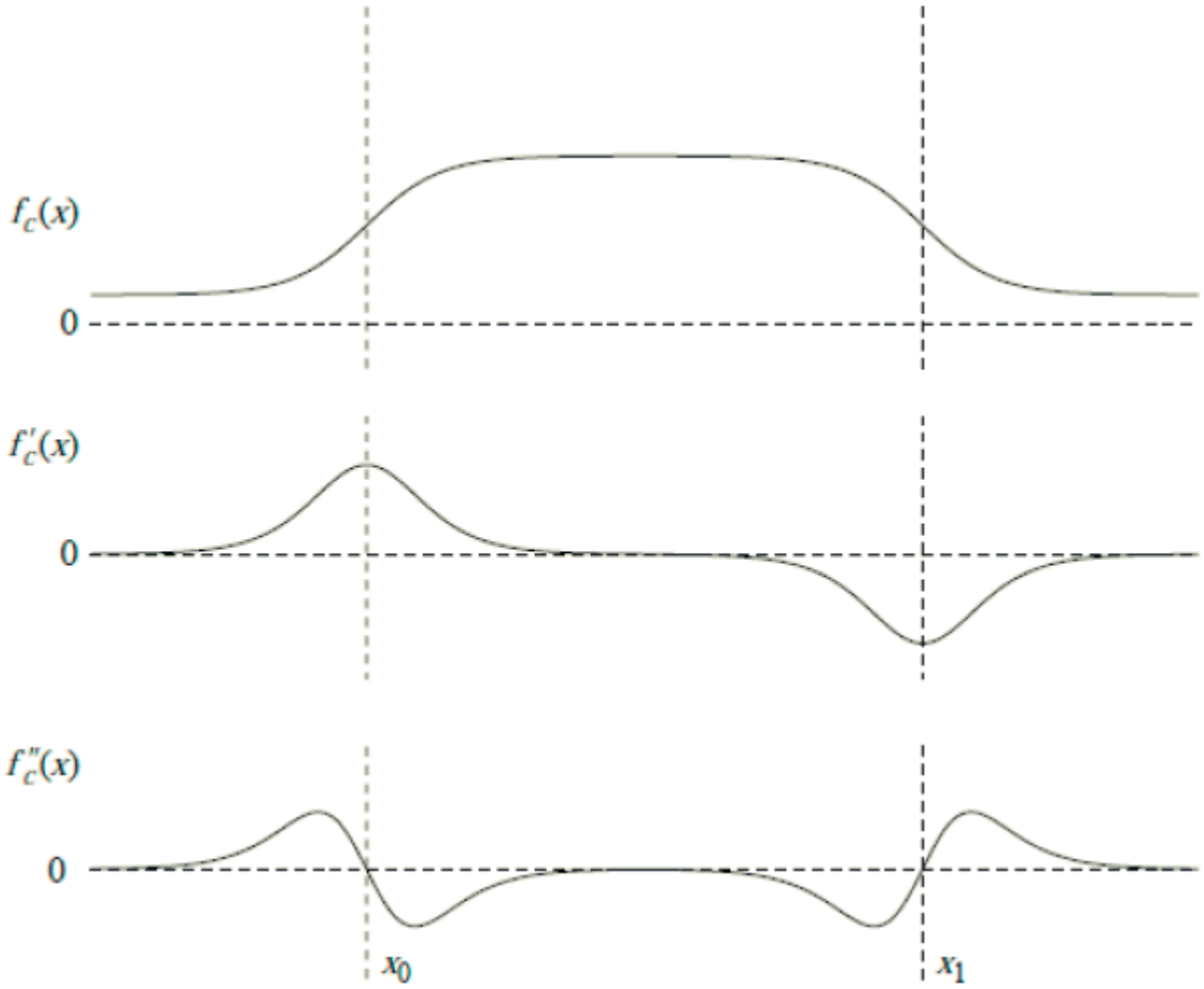


Profil d'intensité



- Par définition, un **contour** est la **frontière** qui sépare deux objets dans une image.
  - *Une discontinuité de l'image*
- Dans notre cas, nous détecterons toutes les lignes marquant des **changements d'intensité** (*pas seulement les contours).*

Rappel sur les dérivées



Rappel sur les dérivées

- Les dérivés d'une fonction numérique sont définies en utilisant la méthode des **différences finies**.

- On obtient une approximation de la dérivée du premier ordre en un point arbitraire x d'une fonction unidimensionnelle $f(x)$ en développant la fonction $f(x + \Delta x)$ en **une série de Taylor** autour de x.

$$f(x + \Delta x) = f(x) + \Delta x \frac{\partial f(x)}{\partial x} + \frac{(\Delta x)^2}{2!} \frac{\partial^2 f(x)}{\partial x^2} + \frac{(\Delta x)^3}{3!} \frac{\partial^3 f(x)}{\partial x^3} + \cdots \quad \textbf{Eq. (1)}$$

$$= \sum_{n=0}^{\infty} \frac{(\Delta x)^n}{n!} \frac{\partial^n f(x)}{\partial x^n}$$

où Δx est la différence entre deux échantillons de f mesurée en unités de pixels.

Rappel sur les dérivées

Si $\Delta \boldsymbol{x} = \mathbf{1}$, alors l'équation précédente devient:

$$f(x+1) = f(x) + \frac{\partial f(x)}{\partial x} + \frac{1}{2!}\frac{\partial^2 f(x)}{\partial x^2} + \frac{1}{3!}\frac{\partial^3 f(x)}{\partial x^3} + \cdots \qquad \textbf{Eq. (2)}$$

$$= \sum_{n=0}^{\infty} \frac{1}{n!}\frac{\partial^n f(x)}{\partial x^n}$$

D'une manière similaire, si $\Delta \boldsymbol{x} = -\mathbf{1}$, on aura:

$$f(x-1) = f(x) - \frac{\partial f(x)}{\partial x} + \frac{1}{2!}\frac{\partial^2 f(x)}{\partial x^2} - \frac{1}{3!}\frac{\partial^3 f(x)}{\partial x^3} + \cdots \qquad \textbf{Eq. (3)}$$

$$= \sum_{n=0}^{\infty} \frac{(-1)^n}{n!}\frac{\partial^n f(x)}{\partial x^n}$$

Dans la suite de ce cours, nous calculons les différences d'intensité en utilisant seulement quelques termes de la série de Taylor.

Rappel sur les dérivées

La **différence avant** est obtenue à partir de l'Eq. (2) comme suit:

$$\frac{\partial f(x)}{\partial x} = f'(x) = f(x+1) - f(x)$$

La **différence arrière** est obtenue à partir de l'Eq. (3) comme suit:

$$\frac{\partial f(x)}{\partial x} = f'(x) = f(x) - f(x-1)$$

et la **différence centrale** est obtenue en soustraction:

$$\frac{\partial f(x)}{\partial x} = f'(x) = \frac{f(x+1) - f(x-1)}{2}$$

- Les termes d'ordres supérieurs de la série que nous n'avons pas utilisés représentent l'erreur entre l'expansion exacte et approximative de la dérivée.
- En général, plus nous utilisons des termes de la série de Taylor pour représenter un dérivé, plus l'approximation sera précise.

Rappel sur les dérivées

La dérivée du second ordre basée sur une différence centrale:

$$\frac{\partial^2 f(x)}{\partial x^2} = f''(x) = f(x+1) - 2f(x) + f(x-1)$$

Pour obtenir **la dérivée centrale du troisième ordre**, nous avons besoin d'un point de plus de chaque côté de x.

$$\frac{\partial^3 f(x)}{\partial x^3} = f'''(x) = \frac{1}{2}\big(f(x+2) - 2f(x+1) + 2f(x-1) - f(x-2)\big)$$

De même **la dérivée centrale du quatrième ordre** après avoir ignoré tous les termes d'ordre supérieur est donnée par:

$$\frac{\partial^4 f(x)}{\partial x^4} = f''''(x) = f(x+2) - 4f(x+1) + 6f(x) - 4f(x-1) + f(x-2)$$

Rappel sur les dérivées

- Le tableau suivant résume les quatre premières dérivées centrales dont nous venons de présenter. Notez la symétrie des coefficients autour du point central.

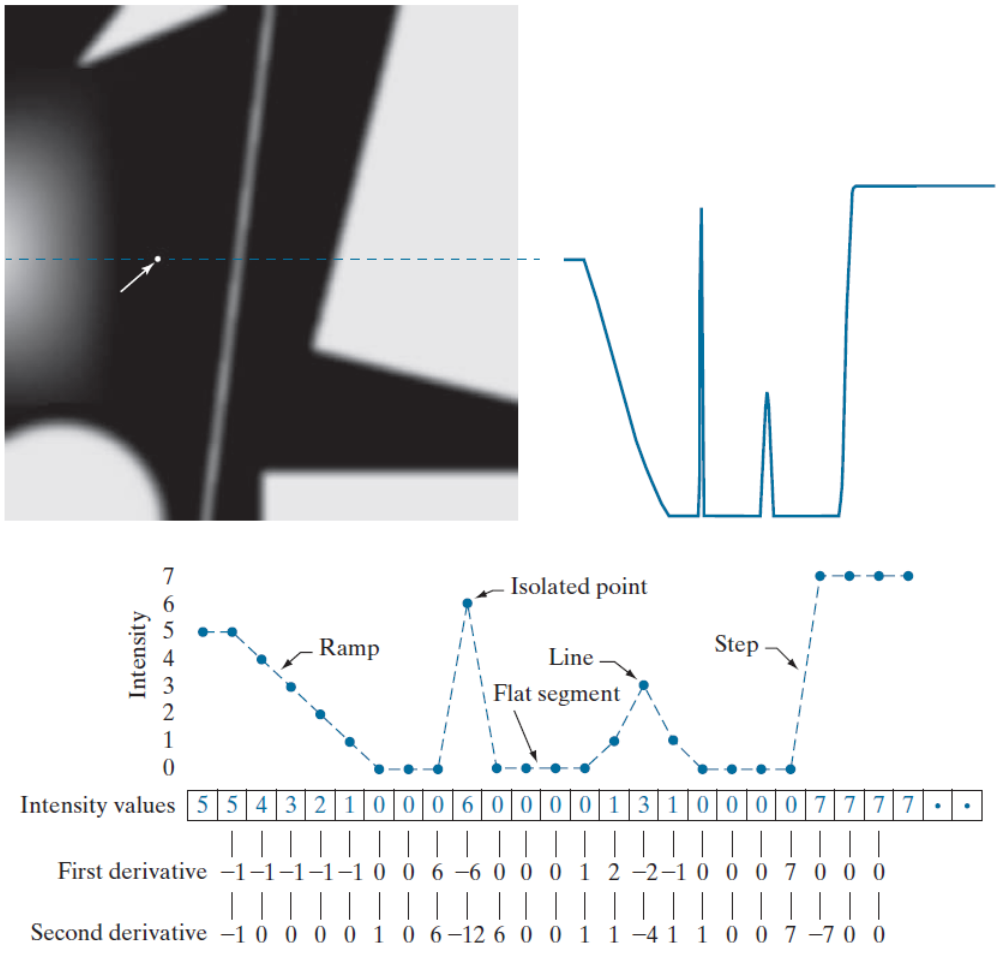
TABLE 10.1
First four central digital derivatives (finite differences) for samples taken uniformly, $\Delta x = 1$ units apart.

	$f(x+2)$	$f(x+1)$	$f(x)$	$f(x-1)$	$f(x-2)$
$2f'(x)$		1	0	−1	
$f''(x)$		1	−2	1	
$2f'''(x)$	1	−2	0	2	−1
$f''''(x)$	1	−4	6	−4	1

Rappel sur les dérivées

a b
c

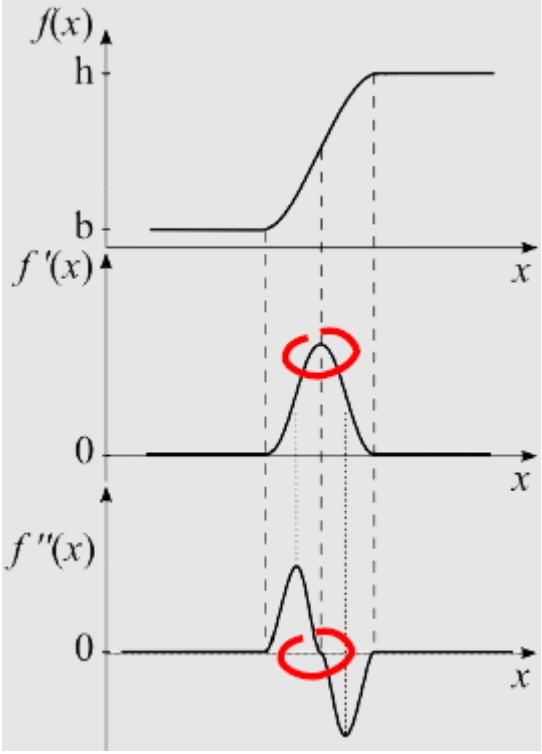
FIGURE 10.2
(a) Image.
(b) Horizontal intensity profile that includes the isolated point indicated by the arrow.
(c) Subsampled profile; the dashes were added for clarity. The numbers in the boxes are the intensity values of the dots shown in the profile. The derivatives were obtained using Eqs. (10-4) for the first derivative and Eq. (10-7) for the second.



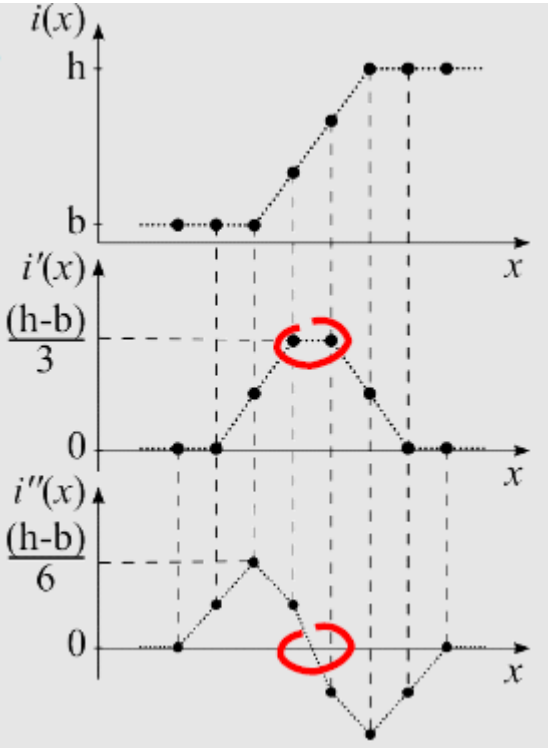
Caractérisation des points contours

Mise en évidence des zones de contours : dérivées première et seconde

Fonctions continues



Fonctions discrètes



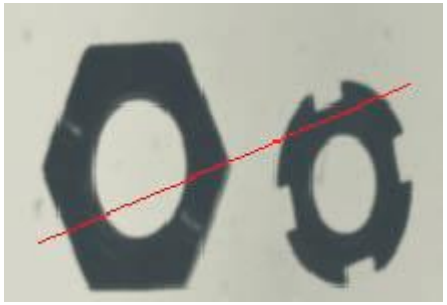
Détection des points contours : utilisation d'un critère de décision

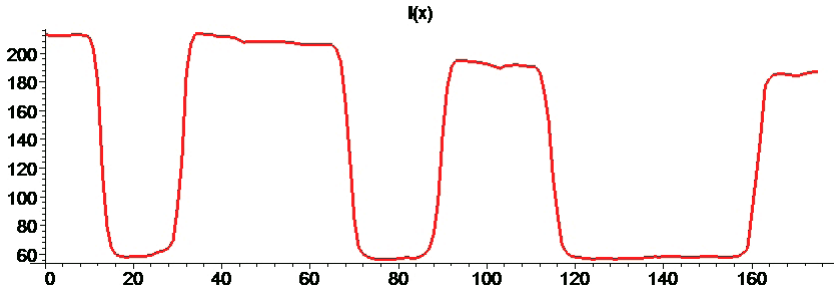
- **Dérivée première : maxima locaux**
- **Dérivée seconde : passages par zéro**

Dérivée d'une image et contour

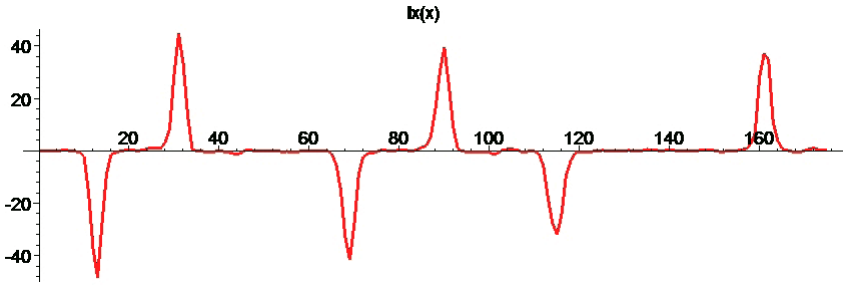
On peut observer qu'un contour correspond généralement à une **discontinuité d'intensité;** Une discontinuité d'intensité correspond à un pique au niveau de la **dérivée d'ordre 1.**

Une coupe des niveaux gris dans une image à deux dimensions





Niveaux de gris le long de la ligne rouge



Dérivée d'ordre 1 des niveaux de gris.
Les contours sont les maxima positifs et les minima négatifs.

Dérivée d'une image et contour

Image 1D f(x)

1ère dérivée f'(x)

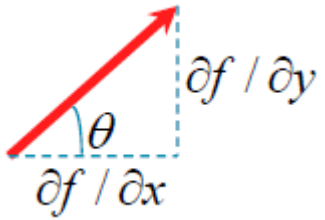
|f'(x)|

Pixels contours:
|f'(x)| > Seuil

Rappel du gradient

- L'outil utilisé pour trouver la force et la direction des contours à un point arbitraire $(\boldsymbol{x}, \boldsymbol{y})$ d'une fonction 2D (image) $\boldsymbol{f}$ est **le gradient**, noté $\boldsymbol{\nabla f}$ et défini comme le vecteur

$$\nabla f(x,y) \equiv \text{grad}\left[f(x,y)\right] \equiv \begin{bmatrix} g_x(x,y) \\ g_y(x,y) \end{bmatrix} = \begin{bmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \end{bmatrix}$$



- **L'amplitude** $\boldsymbol{M(x,y)}$ de ce vecteur de gradient en un point $(\boldsymbol{x}, \boldsymbol{y})$ est donnée par sa norme vectorielle euclidienne:

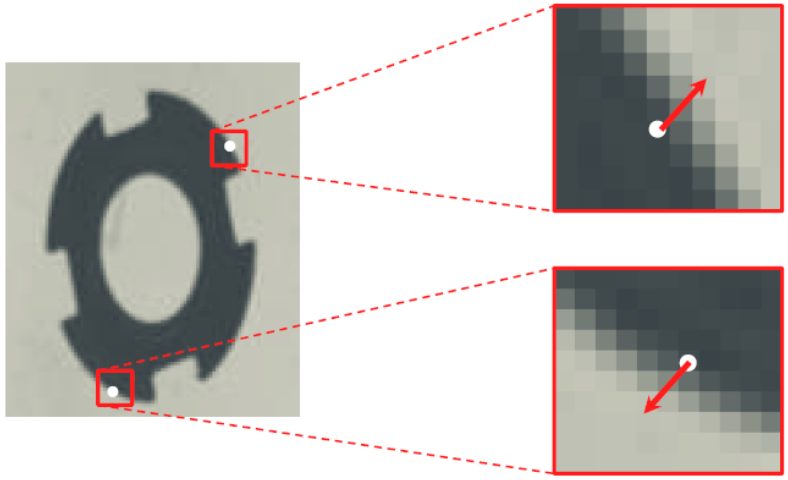
$$M(x,y) = \|\nabla f(x,y)\| = \sqrt{{g_x}^2(x,y) + {g_y}^2(x,y)}$$

- **La direction** du vecteur gradient en un point $(\boldsymbol{x}, \boldsymbol{y})$ est donnée par :

$$\alpha(x,y) = tan^{-1}\left[\frac{g_x(x,y)}{g_y(x,y)}\right]$$

Rappel du gradient

- Le gradient pointe toujours en direction du changement maximal.



- Plus **l'amplitude** du gradient est **forte,** plus la discontinuité est **brusque.**

Opérateur de gradient

- En utilisant les différences avant, nous obtenons:

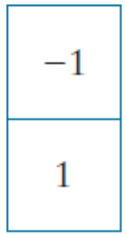
$$g_x(x,y) = \frac{\partial f(x,y)}{\partial x} = f(x+1,y) - f(x,y)$$

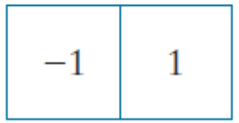
et

$$g_y(x,y) = \frac{\partial f(x,y)}{\partial y} = f(x,y+1) - f(x,y)$$

a b

FIGURE 10.13
1-D kernels used to implement Eqs. (10-19) and (10-20).





Opérateur de gradient

- Nous introduisons trois filtres permettant d'effectuer un calcul de gradient en une seule étape:

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

−1	0		0	−1	
0	1		1	0	

Roberts

−1	−1	−1	−1	0	1
0	0	0	−1	0	1
1	1	1	−1	0	1

Prewitt

−1	−2	−1	−1	0	1
0	0	0	−2	0	2
1	2	1	−1	0	1

Sobel

Opérateur de gradient

- **Roberts**

$$g_x = \frac{\partial f}{\partial x} = (z_9 - z_5)$$
$$g_y = \frac{\partial f}{\partial y} = (z_8 - z_6)$$

- **Prewitt**

$$g_x = \frac{\partial f}{\partial x} = (z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)$$
$$g_y = \frac{\partial f}{\partial y} = \frac{\partial f}{\partial x} = (z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)$$

- **Sobel**

$$g_x = \frac{\partial f}{\partial x} = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$
$$g_y = \frac{\partial f}{\partial y} = \frac{\partial f}{\partial x} = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

Méthode par seuillage de l'amplitude du gradient

- Une approche pour extraire les contours est de faire le seuillage pour l'amplitude du gradient:

1.Calculer l'amplitude :

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

2.Calculer $\boldsymbol{c(x, y)}$ en utilisant un seuil $\boldsymbol{s}$

$$\boldsymbol{c(x,y)} = \begin{cases} \mathbf{1} & \boldsymbol{si} \ \| \boldsymbol{\nabla f} \| > \boldsymbol{s} \\ \mathbf{0} & \boldsymbol{sinon} \end{cases}$$

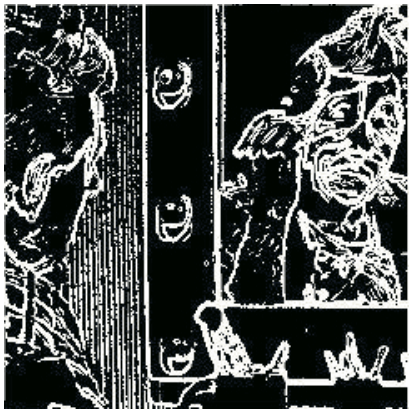
- Pour minimiser le temps de calcul on peut faire le seuillage pour:

$$\| \nabla f \| \approx \left|\frac{\boldsymbol{\partial f}}{\boldsymbol{\partial x}}\right| + \left|\frac{\boldsymbol{\partial f}}{\boldsymbol{\partial y}}\right|$$

Méthode par seuillage de l'amplitude du gradient

Exemple:

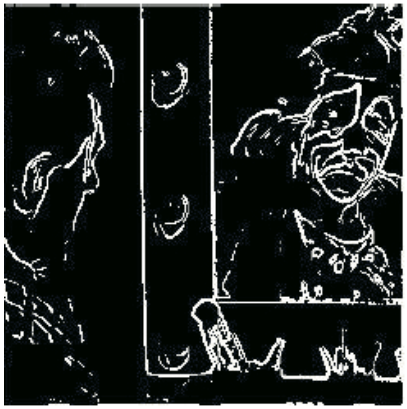
Image



s = 30

s = 40

$||\nabla f||$



s = 60

s =100

Méthode par seuillage de l'amplitude du gradient

- **Inconvénients du seuil avec un simple gradient**
  1. Méthode sensible au **bruit.**
  2. Les contours ne sont pas **filiformes.**
  3. Permet l'apparition de *edge pixels et de edge segments* ***isolés.***

s = 60

s = 60, avec un peu de bruit

Conclusion: cette méthode ne fonctionne bien que pour les **images sans bruit** et avec de **fortes discontinuités.**

Pré-filtrage et seuillage de l'amplitude du gradient

Nous introduisons deux filtres permettant d'effectuer un **filtrage** et un **calcul de gradient** en une seule étape:

Filtre de Perwitt

Filtre « moyenneur» suivi d'un gradient

En x :

$$\begin{pmatrix}-1\\0\\+1\end{pmatrix}\times(1\quad 1\quad 1)/3\rightarrow\begin{pmatrix}-1&-1&-1\\0&0&0\\+1&+1&+1\end{pmatrix}/3$$

En y :

$$\begin{pmatrix}1\\1\\1\end{pmatrix}\times(-1\quad 0\quad +1)/3\rightarrow\begin{pmatrix}-1&0&1\\-1&0&1\\-1&0&1\end{pmatrix}/3$$

Filtre de Sobel

Filtre « quasi-gaussien » suivi d'un gradient

En x :

$$\begin{pmatrix}-1\\0\\+1\end{pmatrix}\times(1\quad 2\quad 1)/4\rightarrow\begin{pmatrix}-1&-2&-1\\0&0&0\\+1&+2&+1\end{pmatrix}/4$$

En y :

$$\begin{pmatrix}1\\2\\1\end{pmatrix}\times(-1\quad 0\quad +1)/4\rightarrow\begin{pmatrix}-1&0&1\\-2&0&2\\-1&0&1\end{pmatrix}/4$$

Pré-filtrage et seuillage de l'amplitude du gradient

- Pour réduire l'effet du bruit: **pré-filtrage**

1. Filtrer l'image d'entrée avec un filtre Gaussien:

$$\boldsymbol{g} = \boldsymbol{f} * \boldsymbol{h}$$

2. Calculer l'amplitude :

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

3. Calculer $\boldsymbol{c}(\boldsymbol{x}, \boldsymbol{y})$ en utilisant un seuil $\boldsymbol{s}$:

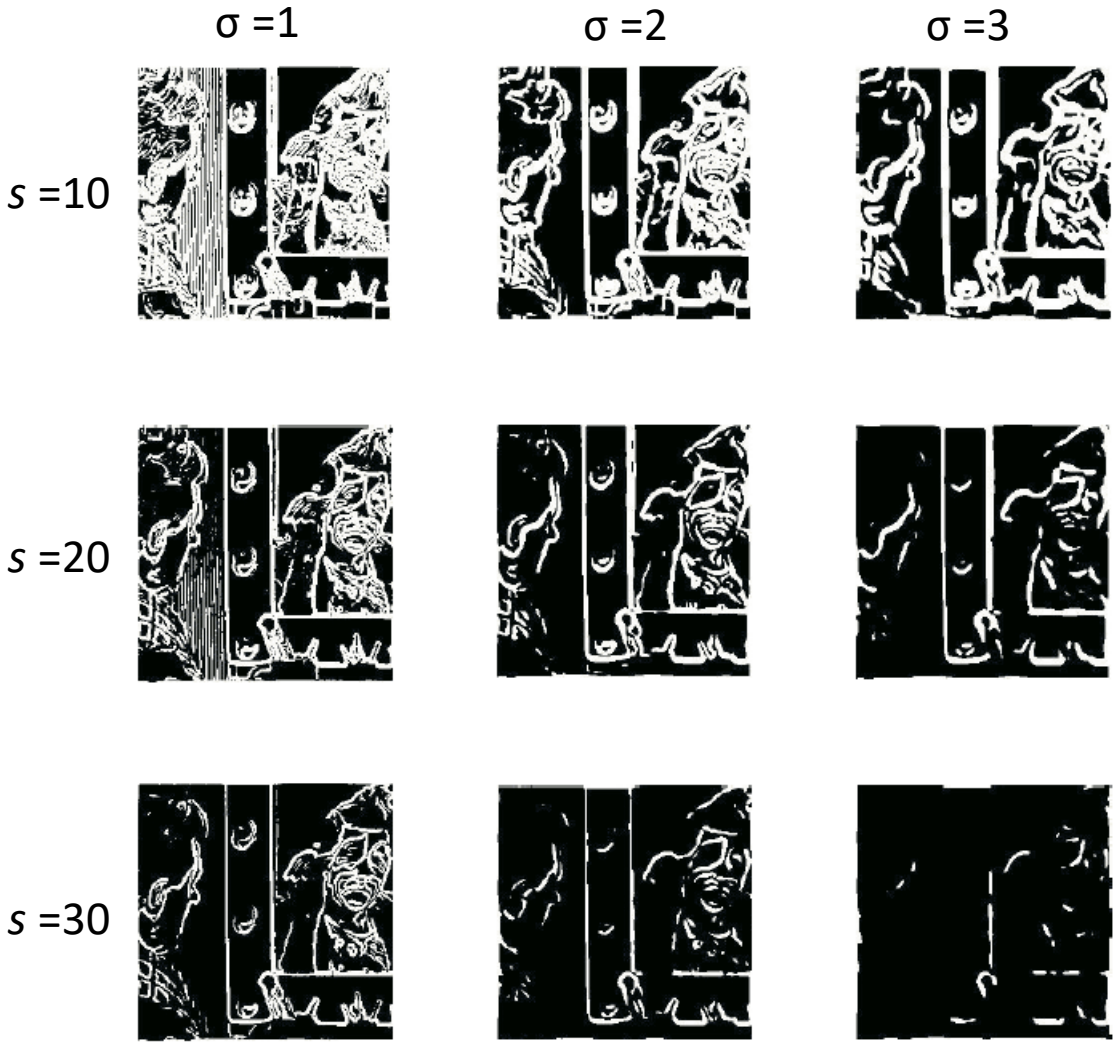
$$\boldsymbol{c}(\boldsymbol{x}, \boldsymbol{y}) = \begin{cases} \mathbf{1} & \boldsymbol{si} \ \| \boldsymbol{\nabla} \mathrm{g} \| > \boldsymbol{s} \\ \mathbf{0} & \boldsymbol{sinon} \end{cases}$$

Méthode utilisant le masque de Sobel

a b
c d

FIGURE 10.16
(a) Image of size 834×1114 pixels, with intensity values scaled to the range [0,1]. (b) $|g_x|$, the component of the gradient in the x-direction, obtained using the Sobel kernel in Fig. 10.14(f) to filter the image. (c) $|g_y|$, obtained using the kernel in Fig. 10.14(g). (d) The gradient image, $|g_x| + |g_y|$.

Pré-filtrage et seuillage de l'amplitude du gradient



Méthode utilisant le masque de Perwitt (Sobel)

Pour réduire les effets du bruit, on peut calculer le gradient de l'image à l'aide d'un filtre de Prewitt (Sobel) *S:*

1. Filtrer l'image d'entrée avec les deux filtres de Prewitt (Sobel):

$$\boldsymbol{g_x = f * S_x}$$
$$\boldsymbol{g_y = f * S_y}$$

2. Calculer le gradient en chaque point de l'image :

$$\| \boldsymbol{\nabla} \mathrm{g} \| = \sqrt{(\boldsymbol{g_x})^2 + \left(\boldsymbol{g_y}\right)^2}$$

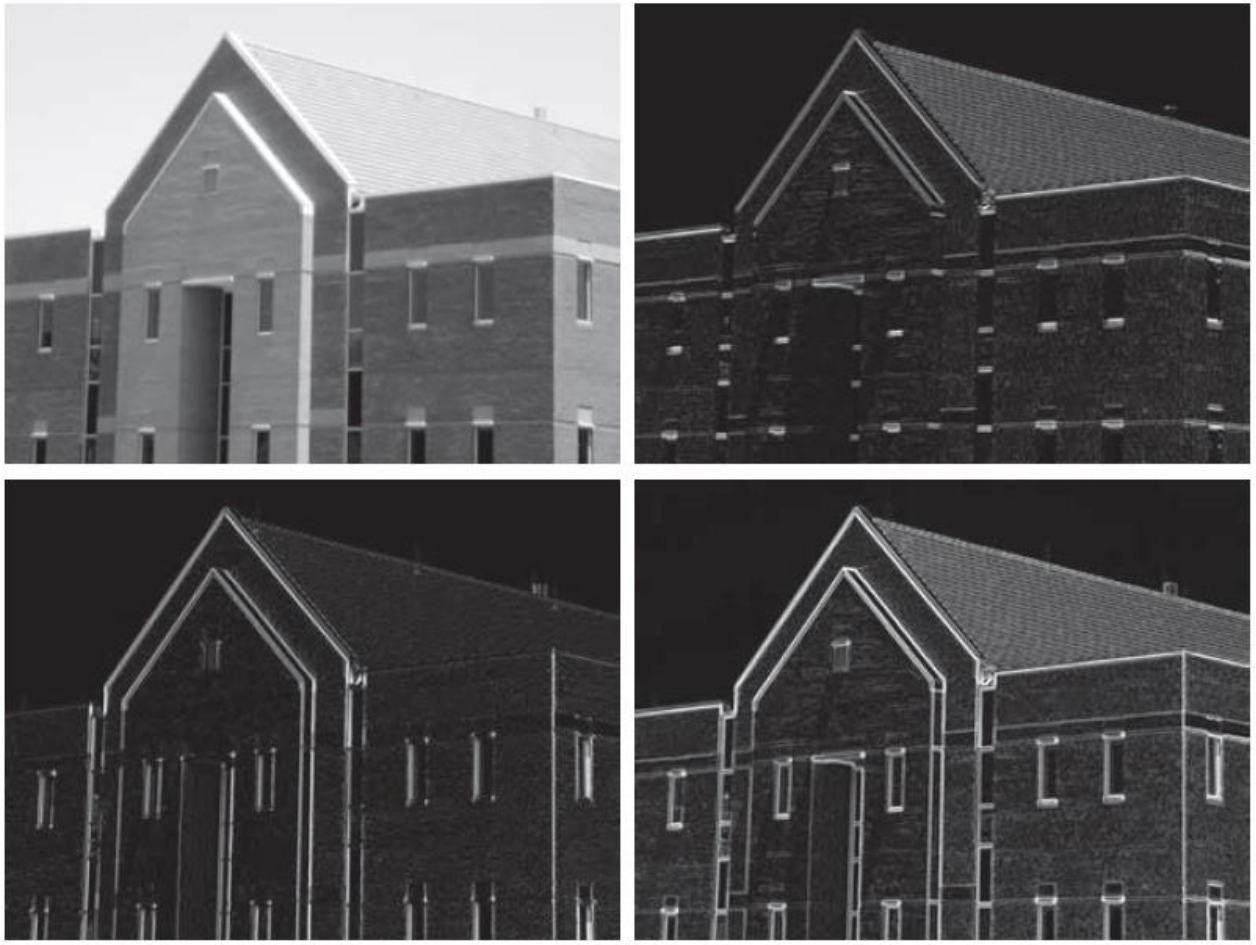
3. Calculer $c(x, y)$ en utilisant un seuil *s*:

$$\boldsymbol{c(x,y)} = \begin{cases} \boldsymbol{1} & \boldsymbol{si} \ \| \boldsymbol{\nabla} \mathrm{g} \| > \boldsymbol{s} \\ \boldsymbol{0} & \boldsymbol{sinon} \end{cases}$$

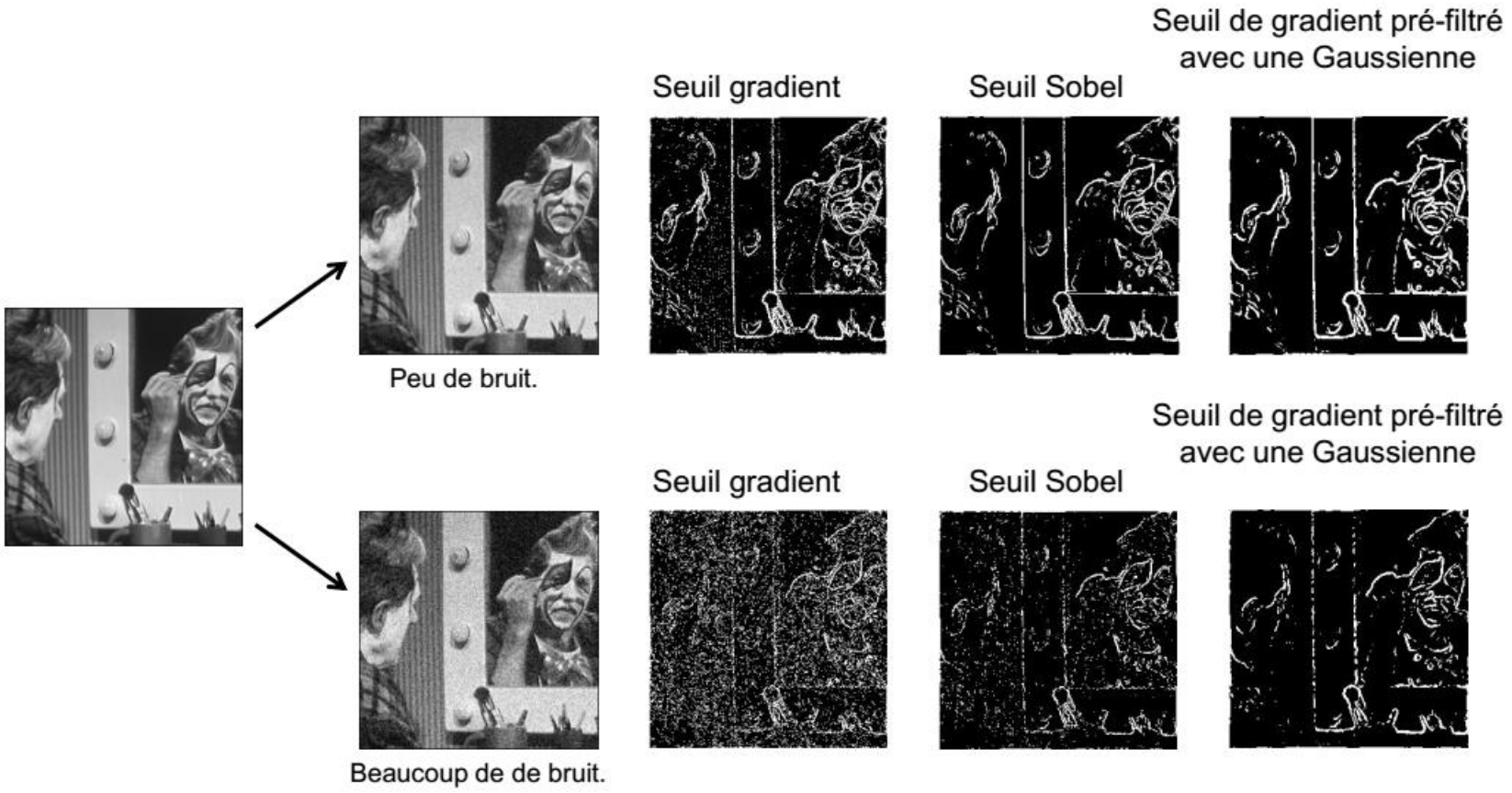
Méthode en appliquant un filtre moyen + le filtre Sobel

a b
c d

FIGURE 10.18
Same sequence as in Fig. 10.16, but with the original image smoothed using a 5×5 averaging kernel prior to edge detection.



Méthode utilisant le masque de Perwitt (Sobel)



Méthode utilisant le masque de Perwitt (Sobel)

En conclusion:

Seuil d'un simple gradient, sans pré-filtrage :

Bon pour les images sans bruit

Seuil d'un gradient obtenu à l'aide du filtre de Sobel :

Bon pour les images peu bruitées

Seuil du gradient d'une image pré-filtrée par un filtre gaussien :

Bon pour les images fortement bruitées

Attention! Un pré-filtrage gaussien tend à arrondir les coins.

Extraction des contours

Inconvénients du seuil d'un simple gradient:

1. Méthode sensible au **bruit**

   Solution : préfiltrer l'image avec un filtre passe-bas

2. Les contours ne sont pas **filiformes**

   Solution : Suppression des non-maximums

3. Permet l'apparition de *edge pixels* et de *edge segments* **isolés**.

   Solution : technique de la mémoire (hystérésis)

C'est ce qu'on appelle communément des **faux positifs**.

Que faire?

Méthode (algorithme) de Canny

C’est un des algorithmes de détection de contours parmi les plus utilisés. Cet algorithme possède cinq étapes:

1. **Lissage:** éliminer le bruit de l’image à l’aide d’un filtre Gaussian.
2. **Calcul des gradients:** Détecter les contours en utilisant la méthode des gradients.
3. **Supprimer les non-maximums:** pour détecter les contours filiformes**.**
4. **Double seuillage:** pour détecter les contour forts.
5. **Seuillage par hystérésis*:*** Éliminer les *edge pixels isolés et les edge segments trop petits**.***

1. Élimination de bruit (Filtre Gaussian)

$$B = \frac{1}{159} \cdot \begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix}$$

(a) Original

(b) Smoothed

2. Calcul du Gradient

$$K_{\mathrm{GX}} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

$$K_{\mathrm{GY}} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

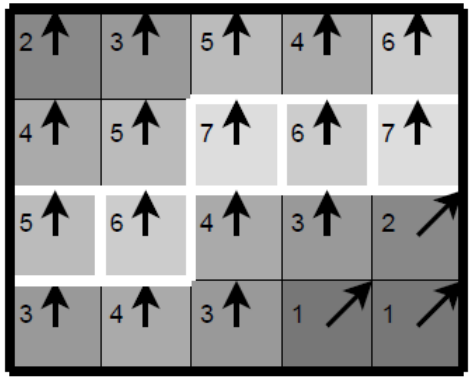
$$|G| = \sqrt{G_{\mathrm{x}}^2 + G_{\mathrm{y}}^2}$$

$$|G| = |G_{\mathrm{x}}| + |G_{\mathrm{y}}|$$

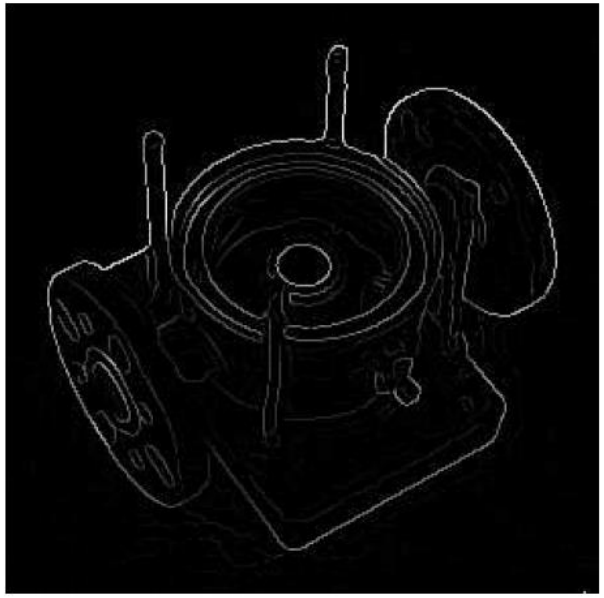
(a) Smoothed

(b) Gradient magnitudes

3. Suppression des non-maximum



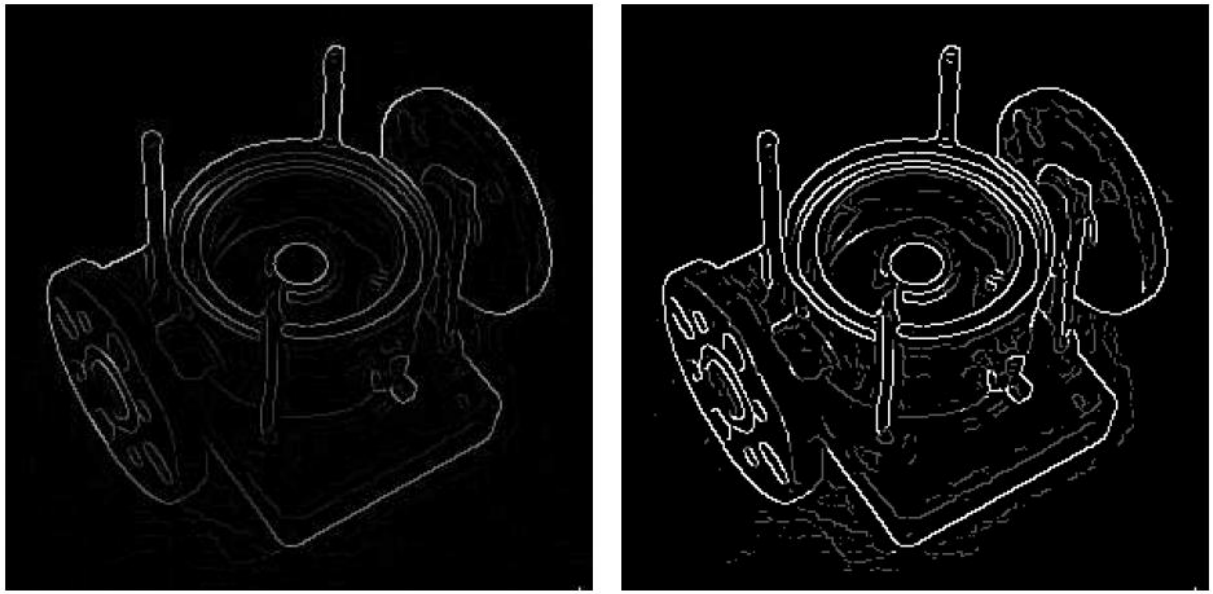
(a) Gradient values



(b) Edges after non-maximum suppression

4. Double seuillage

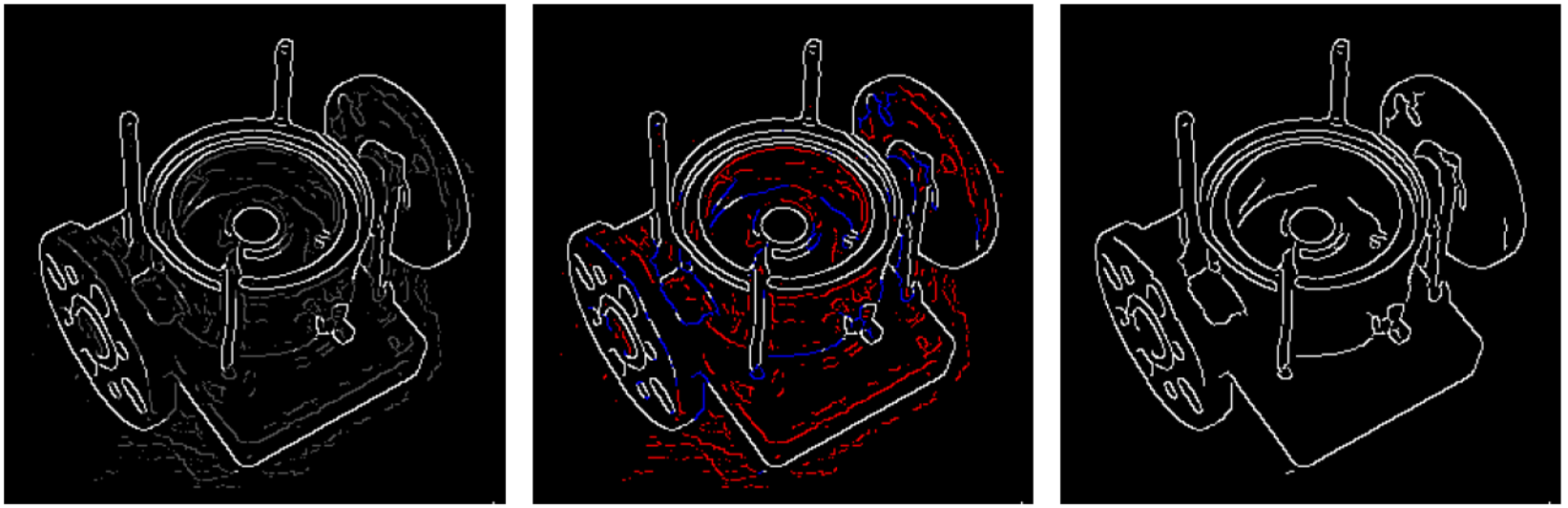
L'effet sur l'image test avec les seuils 20 et 80.



(a) Edges after non-maximum suppression

(b) Double thresholding

5. Seuillage par hystérésis



(a) Double thresholding (b) Edge tracking by hysteresis (c) Final output

Élimine les *edge pixels isolés et les edge segments trop petits.* Pour ce faire, il applique un ***seuillage par hystérésis.***

- ***Blanc*** : contours forts.
- ***Bleu*** : contours faibles connectés à un contour fort.
- ***Rouge*** : autres contours faible.

Seuillage local par hystérésis

But et principe

- Obtenir une image binaire des pixels contours (0=non contour, 1=contour)
- On réalise pour cela un seuillage local basé sur l'hystérésis (« mémoire »)

Algorithme

- **Paramètres :**
  - Image *G* des maxima locaux de la norme du gradient
  - Un seuil bas (S_b) et un seuil haut (S_h), tous deux $\in$ [0, 255]
- **Résultat : image binaire *C* de même taille que *G* .**
- **Principe : à partir des pixels pour lesquels $G(x,y) > S_h$, on « propage » ces contours par connexité tant que $G(x,y) > S_b$.**
- **Détail : pour chaque pixel (x,y),**
  - Si $G(x,y) < S_b$, $C(x,y) = 0$ (le pixel n'est pas contour)
  - Si $G(x,y) > S_h$, $C(x,y) = 1$ (le pixel est un contour)
  - Si $S_b \leq G(x,y) \leq S_h$, $C(x,y) = 1$ s'il est connecté à un autre pixel déjà contour.

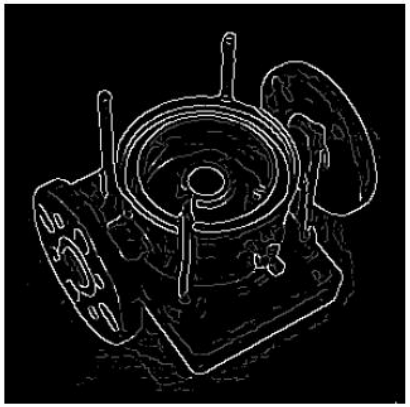
Toutes les étapes du contour Canny

(a) Original

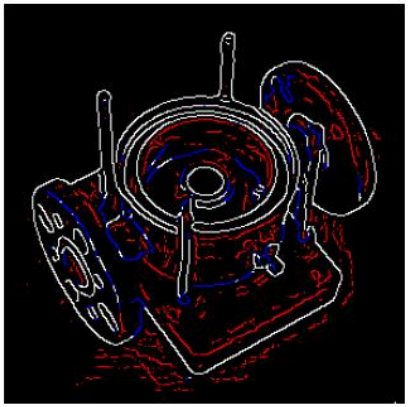
(b) Smoothed

(c) Gradient magnitudes

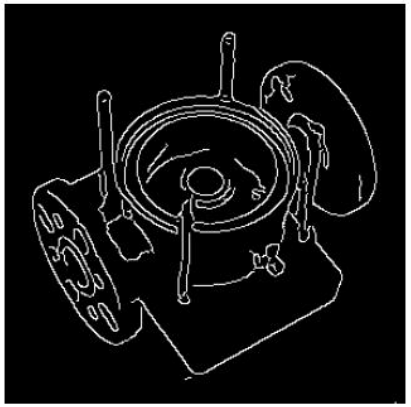
(d) Edges after non-maximum suppression



(e) Double thresholding



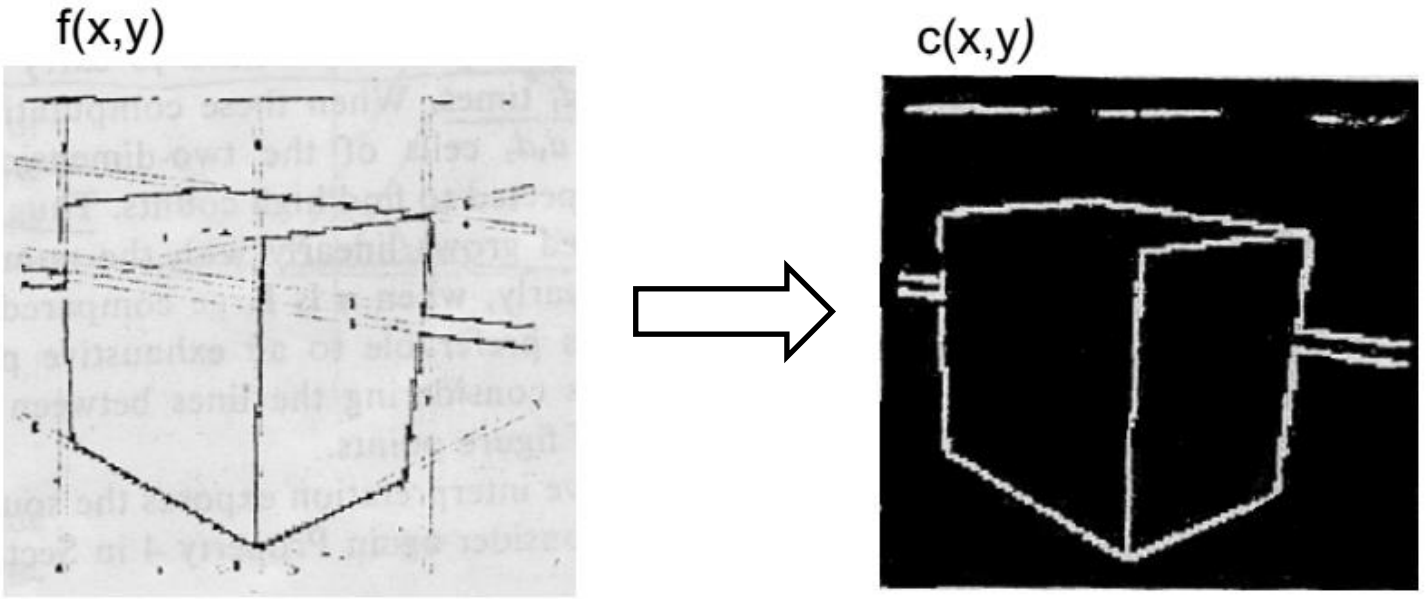
(f) Edge tracking by hysteresis



(g) Final output

2- EXTRACTION DE LIGNES

Introduction



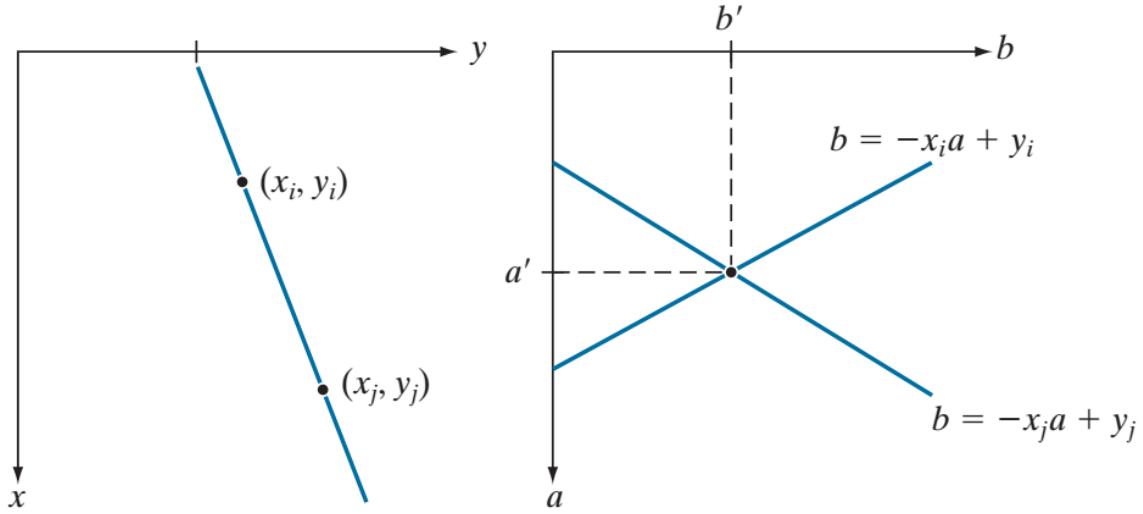
$\boldsymbol{f}(\boldsymbol{x},\boldsymbol{y})$: une image d'entrée (binaire).

$\boldsymbol{c}(\boldsymbol{x},\boldsymbol{y})$: une image binaire, $c(\ i,j\ ) \in$ {lignes, non-lignes}

Introduction

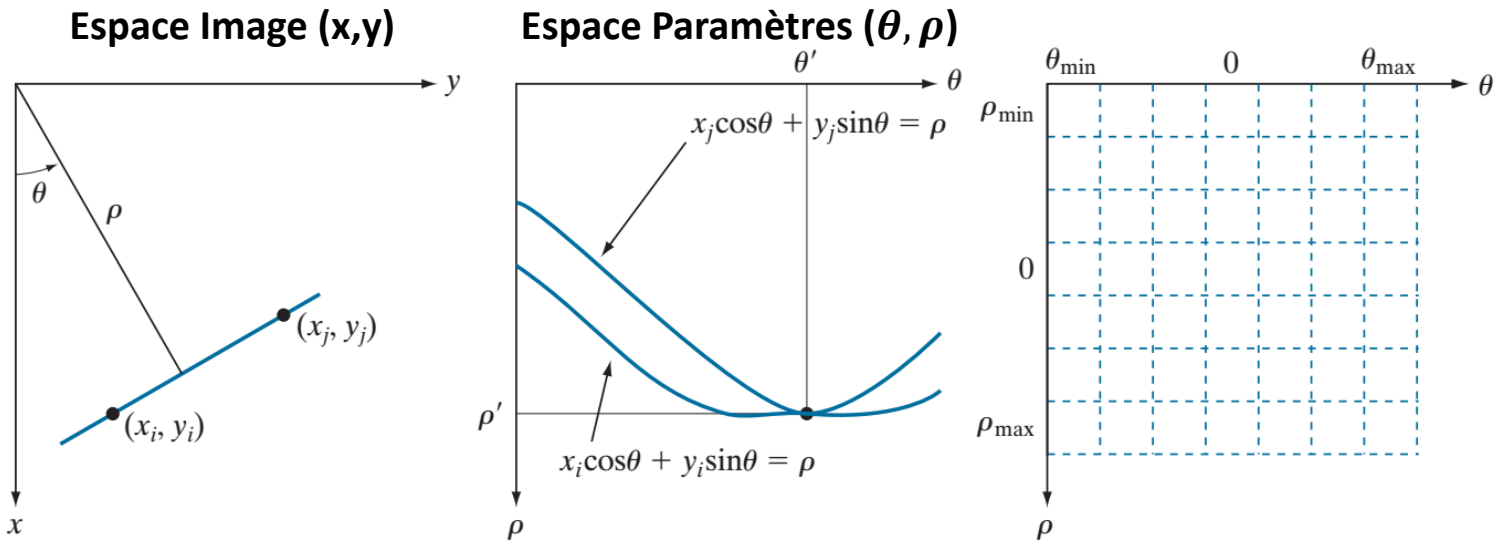
a b

FIGURE 10.28
(a) xy-plane.
(b) Parameter space.



Espace Image (xy) **Espace Paramètres (ab)**

Transformée de Hough



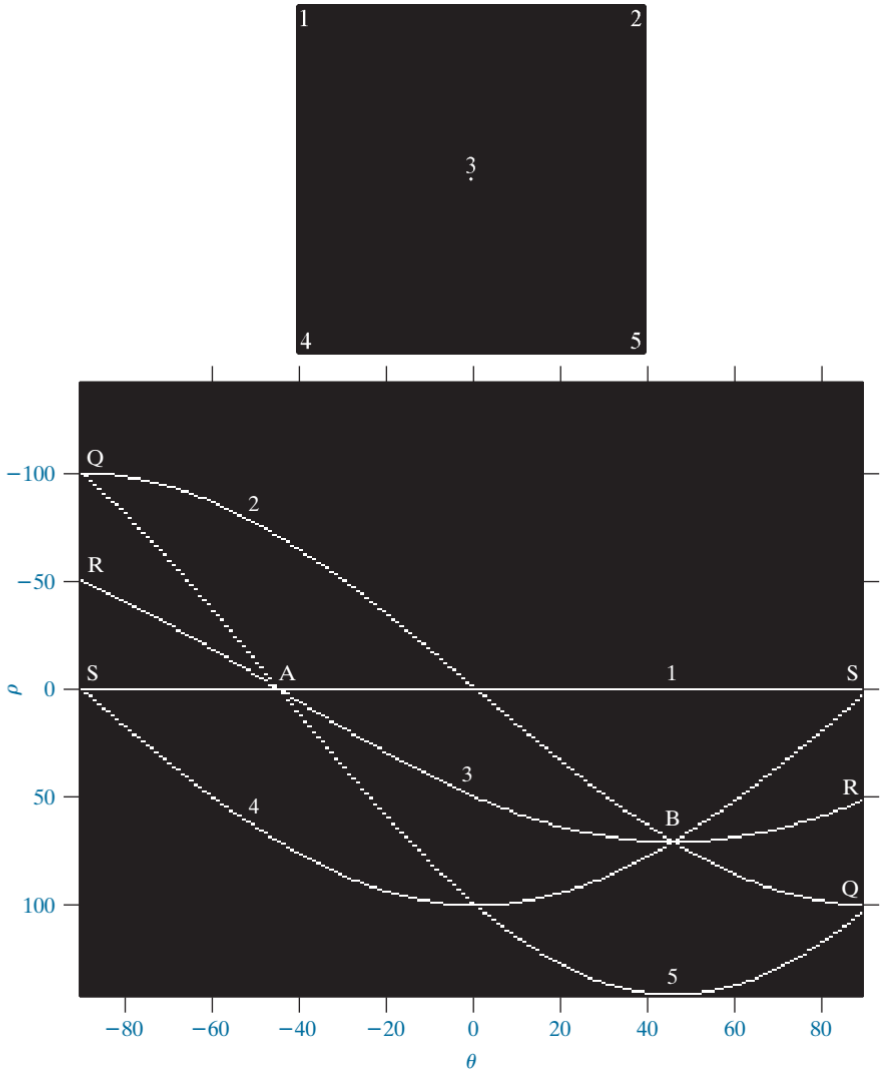
a b c

FIGURE 10.29 (a) (ρ,θ) parameterization of a line in the xy-plane. (b) Sinusoidal curves in the $\rho\theta$-plane;the point of intersection (ρ',θ') corresponds to the line passing through points (x_i,y_i) and (x_j,y_j) in the xy-plane. (c) Division of the $\rho\theta$-plane into accumulator cells.

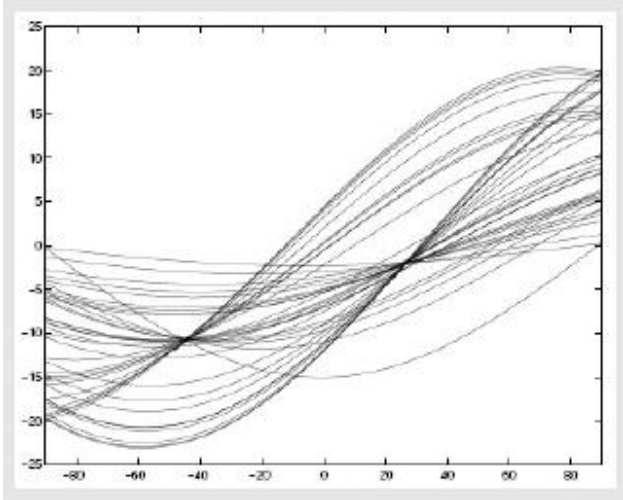
Transformée de Hough

a
b

FIGURE 10.30
(a) Image of size 101×101 pixels, containing five white points (four in the corners and one in the center). (b) Corresponding parameter space.



Transformée de Hough



Système polaire:

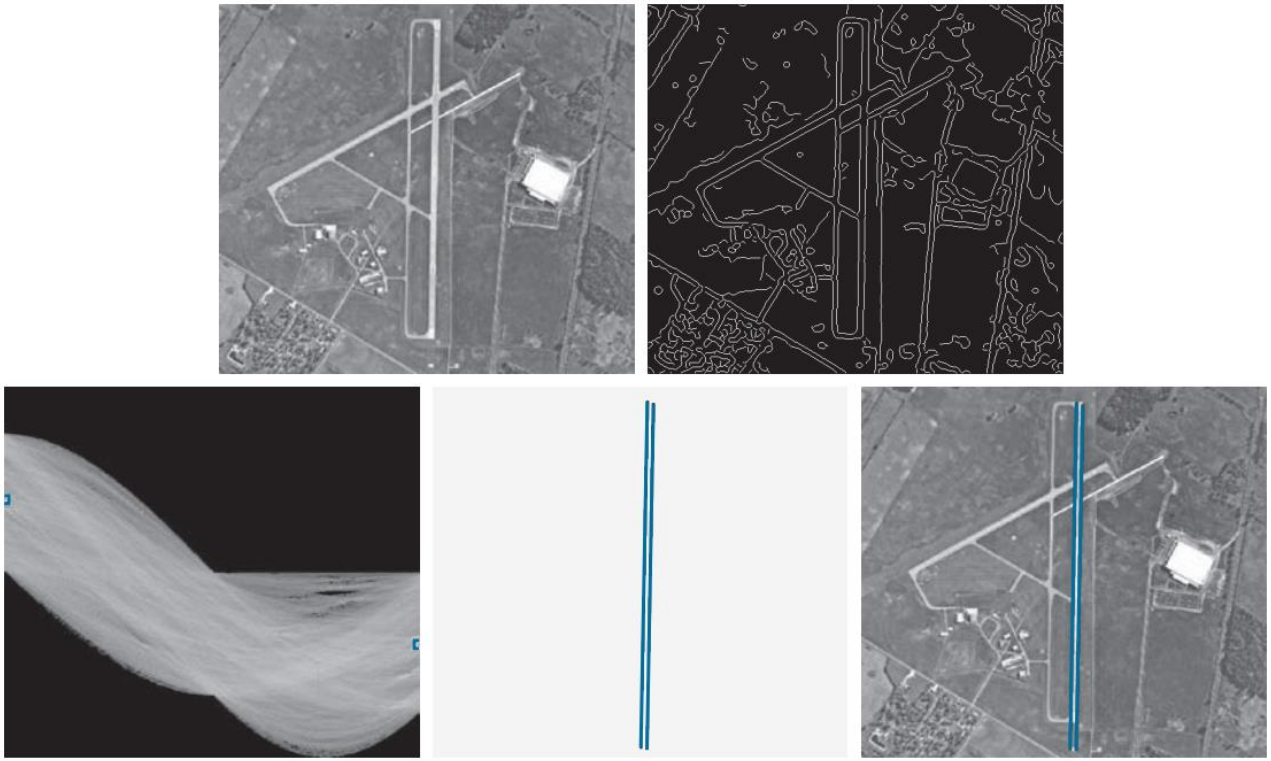
- Chaque point dans **l'espace (x,y)** devient une courbe (une sinusoïde) dans **l'espace (ρ, θ).**
- Chaque point de l'image correspond à une sinusoïde dans **l'espace de paramètre**.
- Les **points d'intersection** dans l'espace de paramètre sont utilisés pour trouver les **droites** dans l'espace image.

Algorithme

1. Appliquer une **détection de contours** (Canny)
2. **Discrétiser** le plan des paramètres (ρ, θ)
3. Initialiser un **accumulateur**
4. **Pour chaque point** sur un contour :
   - 4.1. Déterminer sa courbe image dans **l'espace des paramètres**
   - 4.2. **Incrémenter l'accumulateur** sur les points de cette courbe
5. **Recherche de maxima** → paramètres

Détection de lignes

Exemple:



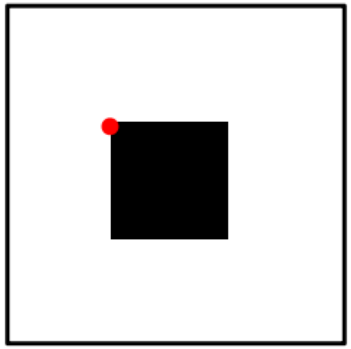
a b
c d e

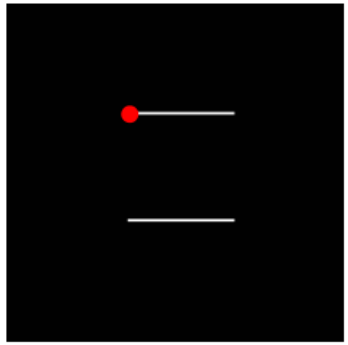
FIGURE 10.31 (a) A 502×564 aerial image of an airport. (b) Edge map obtained using Canny's algorithm. (c) Hough parameter space (the boxes highlight the points associated with long vertical lines). (d) Lines in the image plane corresponding to the points highlighted by the boxes. (e) Lines superimposed on the original image.

3-EXTRACTION DE COINS

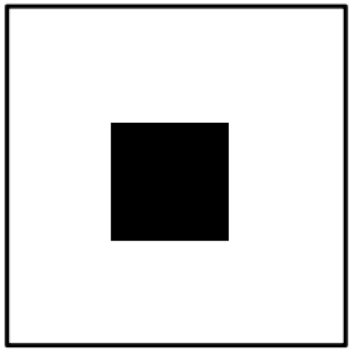
Introduction

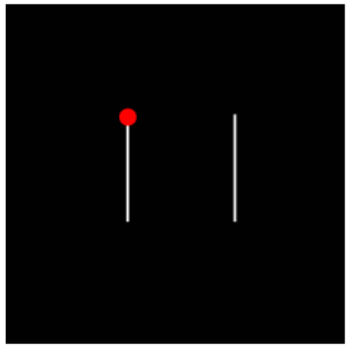
On sait qu'une dérivée en X permet de localiser des contours **horizontaux**





On sait qu'une dérivée en Y permet de localiser des contours **verticaux**





Détecteur de Harris

Pour Harris, un coin est une zone de l'image qui réagit fortement à la dérivée en X **ET** à la dérivée en Y.

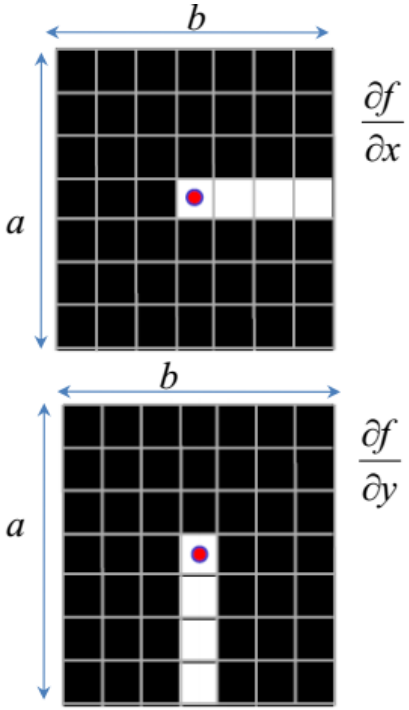
Pour évaluer dans quelle mesure un pixel situé à la position (x, y) réagit aux deux dérivées, Harris calcule **la matrice Hessienne** suivante:

$$H = \begin{pmatrix} A & B \\ B & C \end{pmatrix}$$

$$A = \sum_{i=x-a/2}^{x+a/2} \sum_{j=y-b/2}^{y+b/2} \frac{\partial f(i,j)}{\partial x} \frac{\partial f(i,j)}{\partial x}$$

$$B = \sum_{i=x-a/2}^{x+a/2} \sum_{j=y-b/2}^{y+b/2} \frac{\partial f(i,j)}{\partial x} \frac{\partial f(i,j)}{\partial y}$$

$$C = \sum_{i=x-a/2}^{x+a/2} \sum_{j=y-b/2}^{y+b/2} \frac{\partial f(i,j)}{\partial y} \frac{\partial f(i,j)}{\partial y}$$



Détecteur de Harris

- Un coin est caractérisé par une grande variation de $\boldsymbol{H}$ dans toutes les directions du vecteur (x, y). En analysant **les valeurs propres** de $\boldsymbol{H}$, cette caractérisation peut être exprimée de la manière suivante:

- Une fois la matrice Hessienne calculée, on extrait deux valeurs propres de la matrice **H** ($\boldsymbol{\lambda_1}$ et $\boldsymbol{\lambda_2}$) comme suit:

$$\boldsymbol{det(H - \lambda I) = 0}$$

$$\boldsymbol{\Rightarrow det\left(\begin{pmatrix} A & B \\ B & C \end{pmatrix} - \lambda \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\right) = 0}$$

$$\boldsymbol{\Rightarrow det\left(\begin{pmatrix} A - \lambda & B \\ B & C - \lambda \end{pmatrix}\right) = 0}$$

$$\boldsymbol{\Rightarrow (A - \lambda)(C - \lambda) - B^2 = 0}$$ **(Polynôme caractéristique)**

- Soit $\boldsymbol{\lambda_1}$ et $\boldsymbol{\lambda_2}$ les solutions de ce polynôme ($\boldsymbol{\lambda_1 > \lambda_2}$).

Détecteur de Harris

- Une fois $\boldsymbol{\lambda_1}$ et $\boldsymbol{\lambda_2}$ calculées ($\boldsymbol{\lambda_1} > \boldsymbol{\lambda_2}$), on peut conclure que :
  - Si $\boldsymbol{\lambda_1 \approx 0}$ **et** $\boldsymbol{\lambda_2 \approx 0}$ alors le point (x, y) est sur **une région uniforme**.
  - Si $\boldsymbol{\lambda_1 \gg 0}$ **et** $\boldsymbol{\lambda_2 \approx 0}$ alors le point (x, y) est sur **un contour**.
  - Si $\boldsymbol{\lambda_1 \gg 0}$ **et** $\boldsymbol{\lambda_2 \gg 0}$ alors le point (x, y) est sur **un coin**.
- Pour simplifier les calculs (et ainsi éviter de calculer les racines du polynôme caractéristique), Harris suggère de calculer le terme «$\boldsymbol{M_c}$» à chaque pixel:

$$\boldsymbol{M_c = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2}$$

$$\boldsymbol{M_c = det(H) - k\big(trace(H)\big)^2}$$

  - Si $\boldsymbol{M_c \approx 0}$ alors le point (x,y) est **une région uniforme**.
  - Si $\boldsymbol{M_c \ll 0}$ alors le point (x,y) est **un contours**,
  - Si $\boldsymbol{M_c \gg 0}$ alors le point (x,y) est **un coins**,

Avec

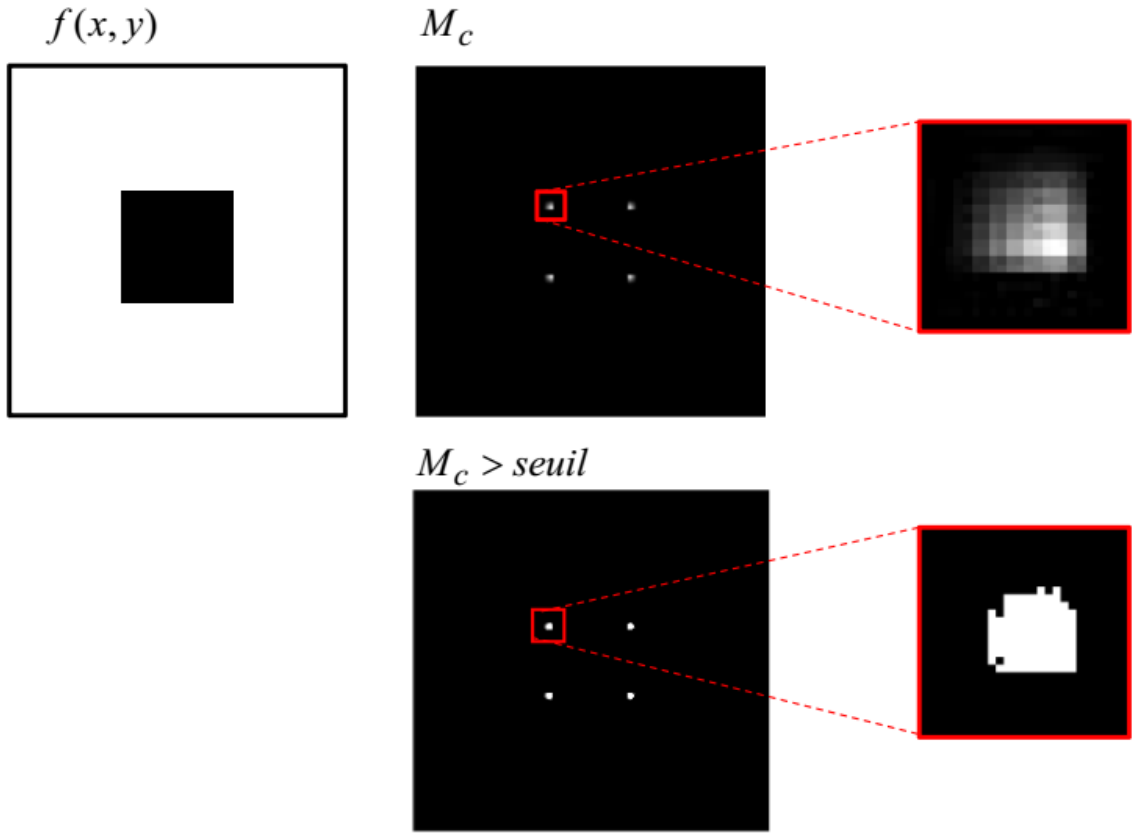
$$\boldsymbol{trace(H) = A + C}$$

$$\boldsymbol{det(H) = AC - B^2}$$

- $\boldsymbol{k}$ prend généralement une valeur entre 0.02 et 0.1.

Détecteur de Harris

Exemple:



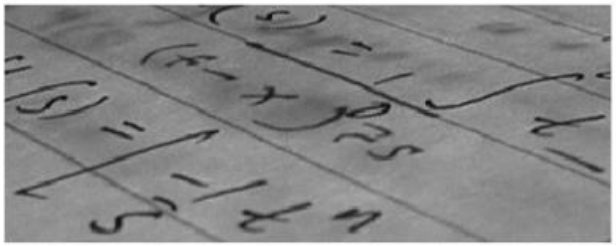
Détecteur de Harris

a b c
d e f

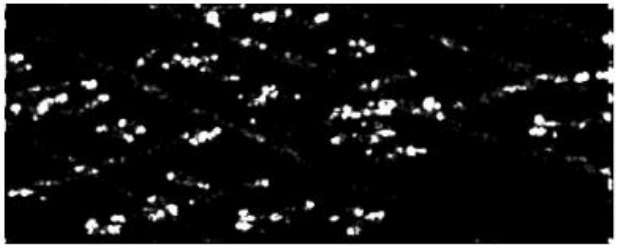
FIGURE 11.49 600×600 image of a building. (b) Result of applying the HS corner detector with $k = 0.04$ and $T = 0.01$ (the default values in our implementation). Numerous irrelevant corners were detected. (c) Result using $k = 0.249$ and the default value for T. (d) Result using $k = 0.17$ and $T = 0.05$. (e) Result using the default value for k and $T = 0.05$. (f) Result using the default value of k and $T = 0.07$.

Détecteur de Harris

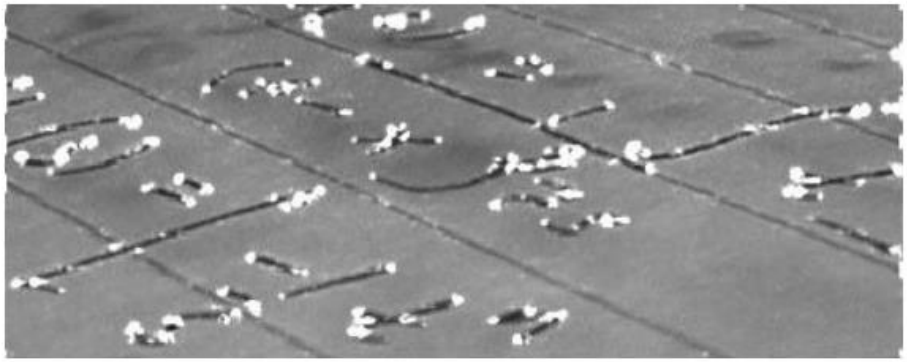
$f(x,y)$



M_c



$f(x,y)+M_c$



Chapitre suivant

Chapitre 04
Restauration et reconstruction d'image

Références:

1. M. S. Allili. Eléments Avancés d'Analyse d'Images (Cours de 2e cycle). Université du Québec en Outaouais (UQO), Québec, Canada. Hivers 2014.
2. R. C. Gonzalez and R. E. Woods. Digital image processing. Pearson Education. 3rd Edition. 2008.
3. R. C. Gonzalez and R. E. Woods. Digital image processing. Pearson Education. 4th Edition. 2018.
4. R. C. Gonzalez, R. E. Woods, and S. L. Eddins. Digital image processing using Matlab. Gatesmark Publishing. 2nd Edition. 2009.