Filtrage dans le domaine spatial

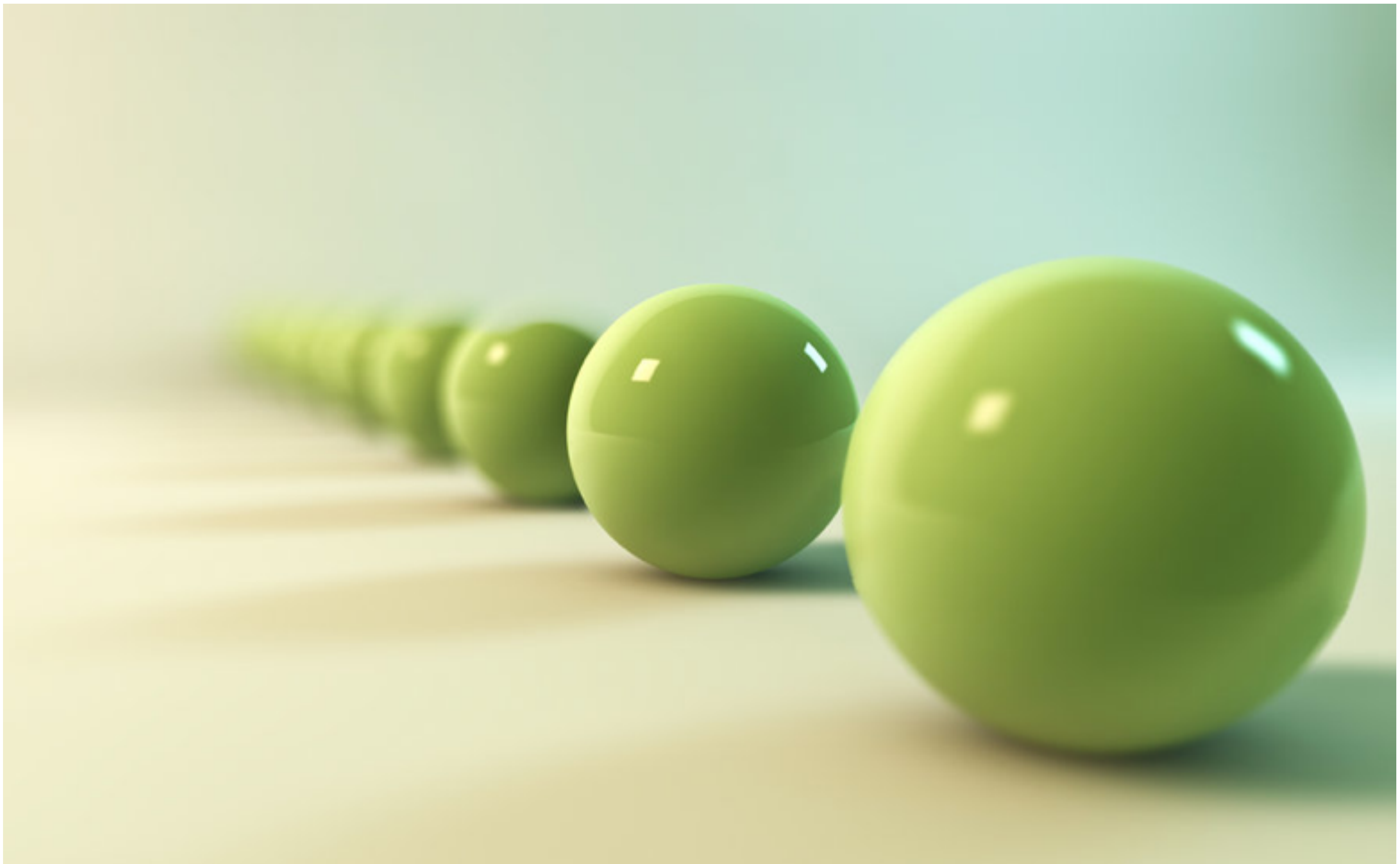


Image: “henriksn”, deviantart

Photographie Algorithmique, H2015

Jean-François Lalonde

Filtrage d’images

- Filtrage: fonction d’un pixel et de ses voisins
- Très important!
  - Modifier l’image
    - Réduire le bruit, re-dimensionner, contraste, etc.
  - Extraire de l’information
    - Texture, edges, distinctive points, etc.
  - Detecter des formes
    - “template matching”

Exemple: boîte

$$g[\cdot\,,\cdot\,]$$

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Slide credit: David Lowe (UBC)

Filtrage

$$g[\cdot,\cdot]\ \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

$$f[.,.]$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$$h[.,.]$$

$$h[m,n] = \sum_{k,l} g[k,l]\, f[m+k, n+l]$$

Credit: S. Seitz

Filtrage

$$g[\cdot,\cdot]\ \frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

	0	10							

$$h[m,n]=\sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Credit: S. Seitz

Filtrage

$g[\cdot,\cdot]\ \frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

	0	10	20						

$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Credit: S. Seitz

Filtrage

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

	0	10	20	30					

$$h[m,n]=\sum_{k,l} g[k,l]\, f[m+k,n+l]$$

Credit: S. Seitz

Filtrage

$g[\cdot,\cdot]\ \frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

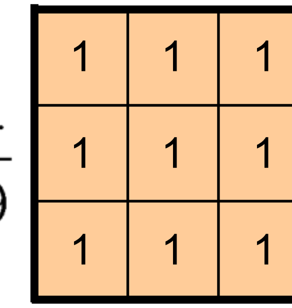
	0	10	20	30					
				?					

$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

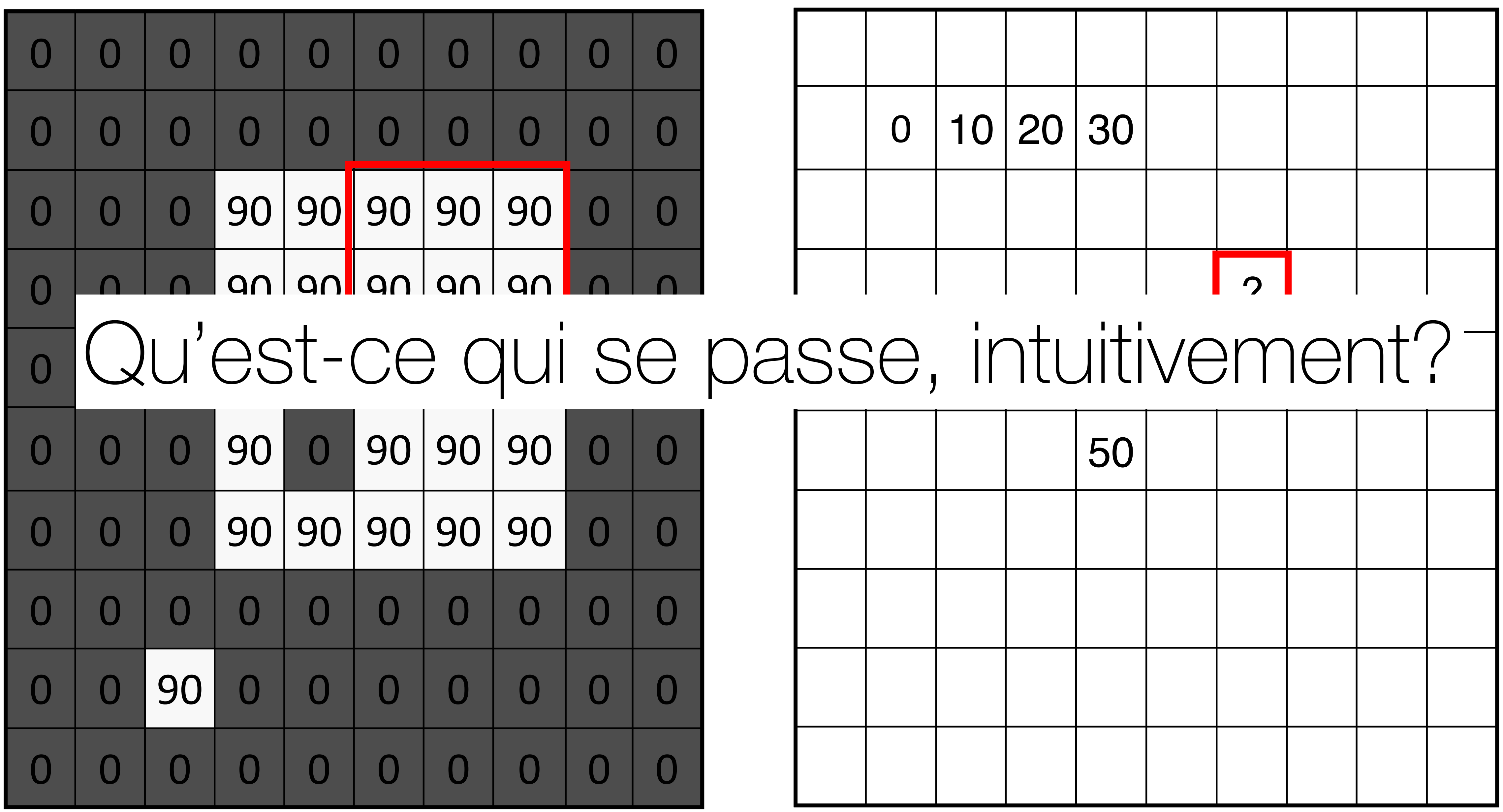
Credit: S. Seitz

Filtrage

$g[\cdot,\cdot]\ \frac{1}{9}$



$f[.,.]$ $h[.,.]$



Qu'est-ce qui se passe, intuitivement?

$$h[m,n]=\sum_{k,l} g[k,l]\, f[m+k,n+l]$$

Credit: S. Seitz

Filtrage

$$g[\cdot\,,\cdot\,] \quad \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

Credit: S. Seitz

Filtrage “en boîte” (box filter)

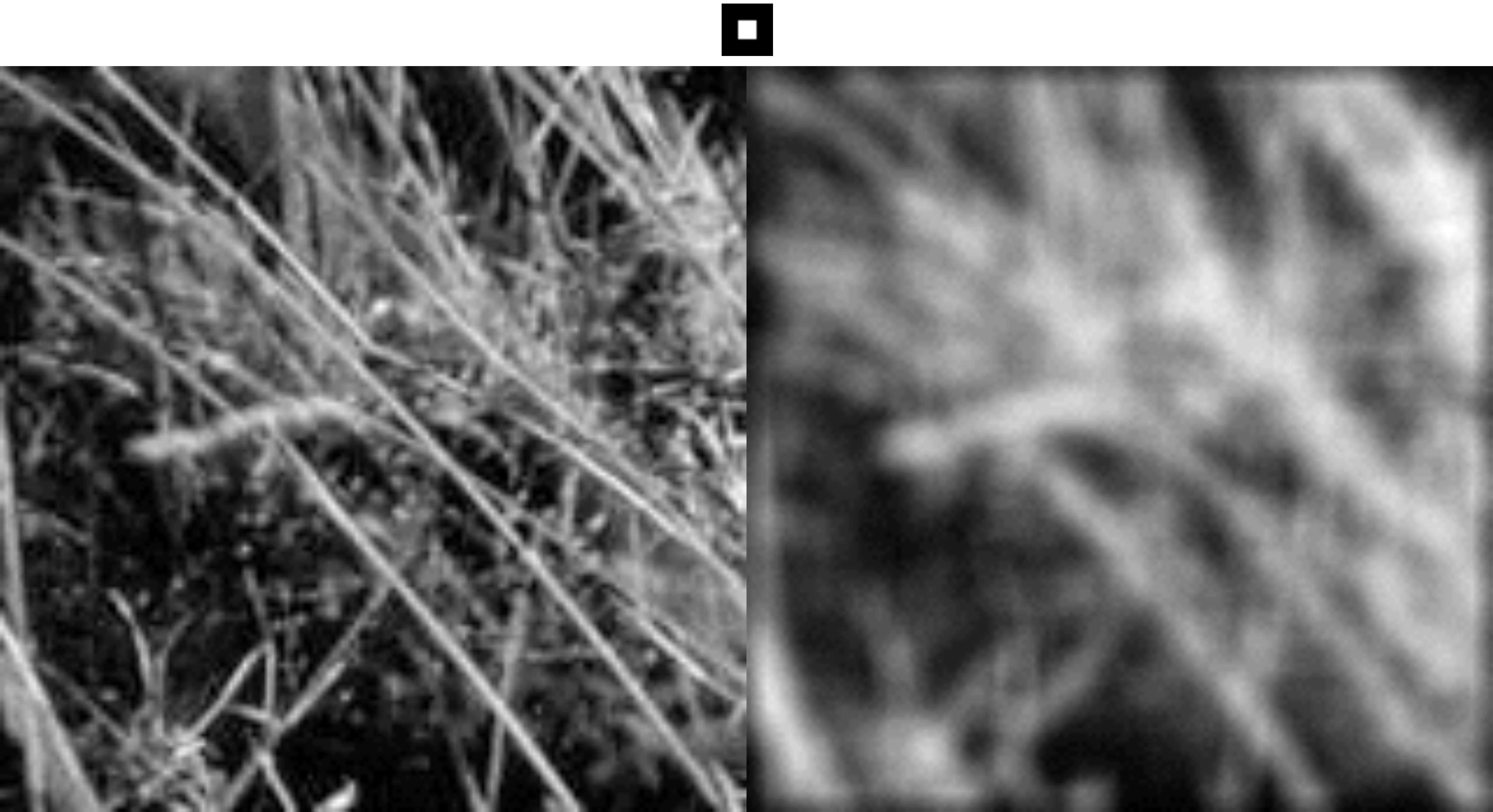
- Remplace chaque pixel par la moyenne de son voisinage
- On adoucit l’image (enlève les hautes fréquences)

$$g[\cdot\,,\cdot\,]$$

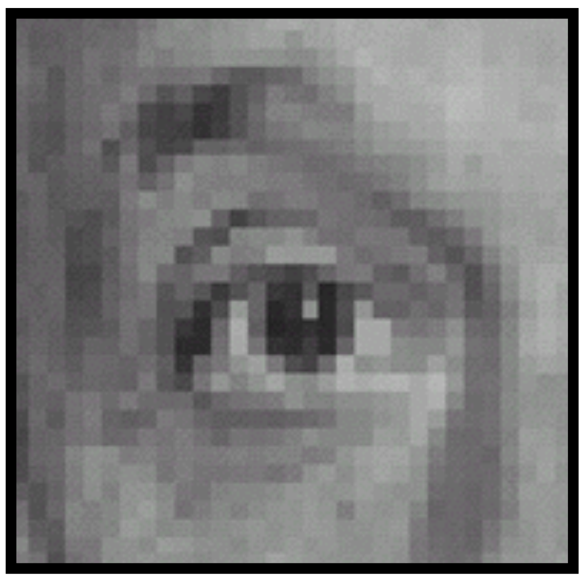
$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Slide credit: David Lowe (UBC)

“Atténuer” l’image avec le filtre boîte



Petite pratique avec les filtres linéaires



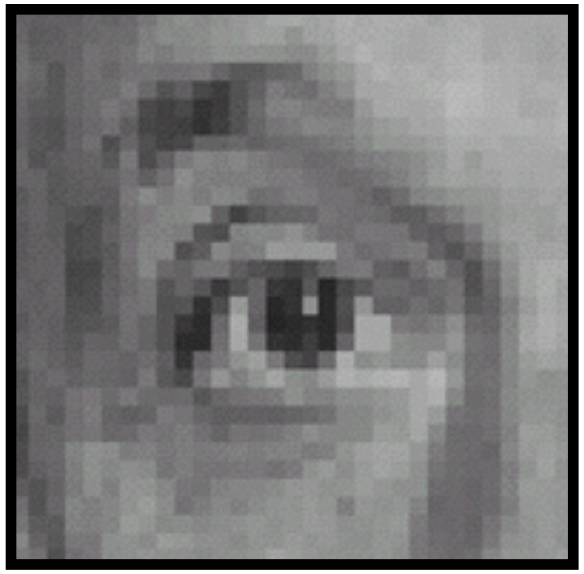
Originale

0	0	0
0	1	0
0	0	0

?

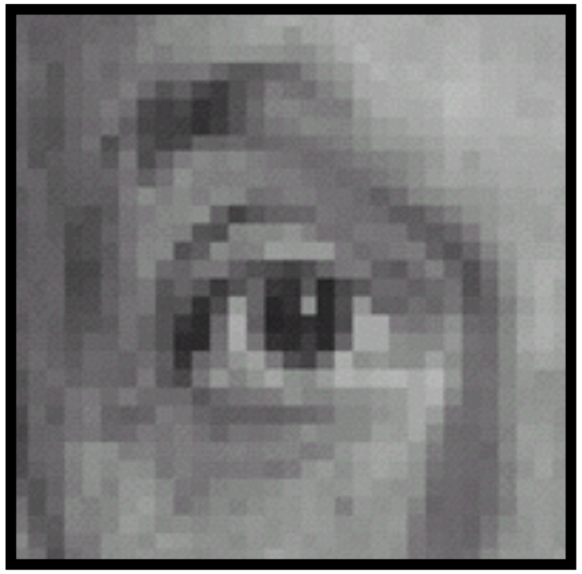
Source: D. Lowe

Petite pratique avec les filtres linéaires



Originale

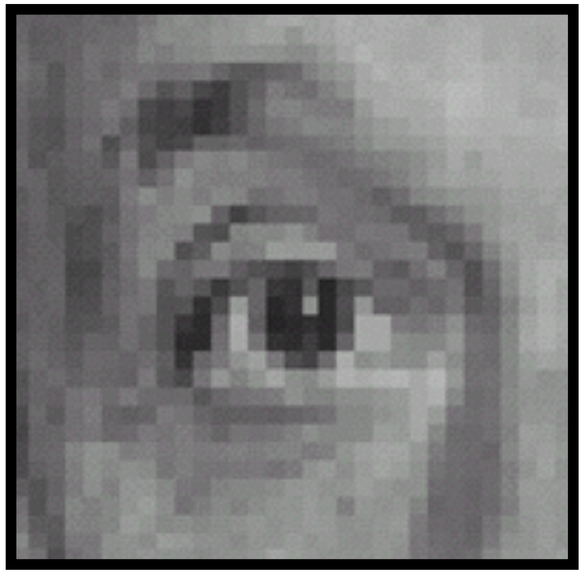
0	0	0
0	1	0
0	0	0



Résultante
(identique!)

Source: D. Lowe

Petite pratique avec les filtres linéaires



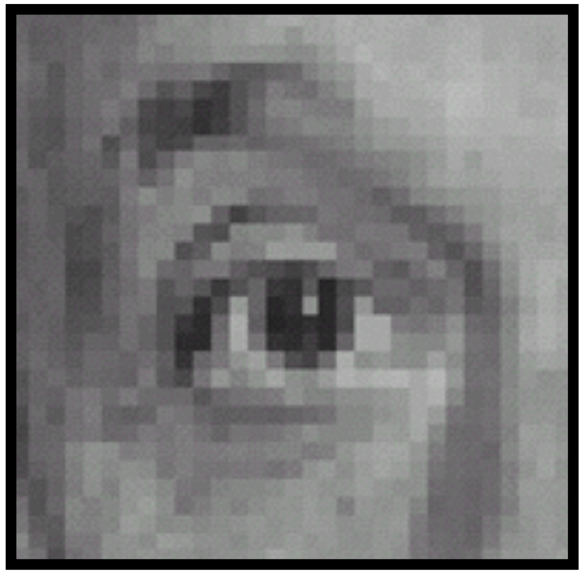
Originale

0	0	0
0	0	1
0	0	0

?

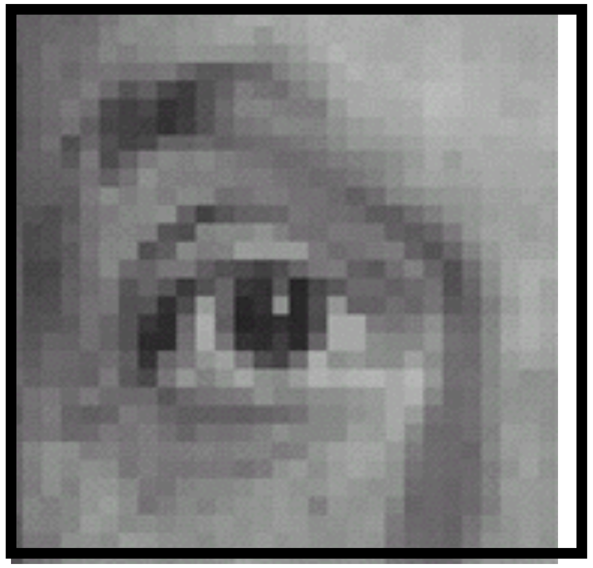
Source: D. Lowe

Petite pratique avec les filtres linéaires



Originale

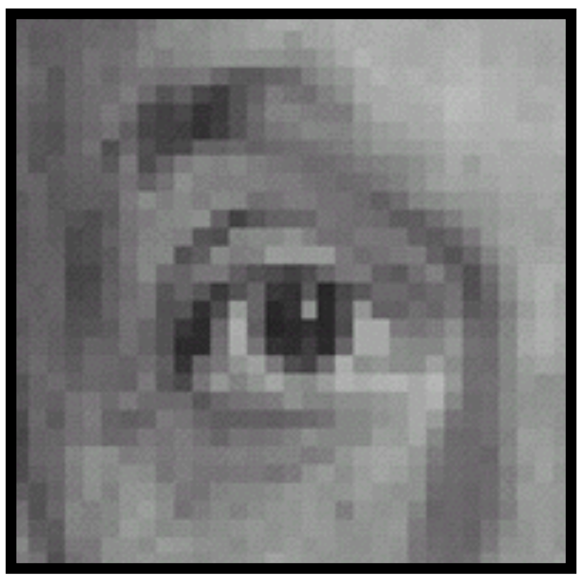
0	0	0
0	0	1
0	0	0



À gauche de 1 pixel

Source: D. Lowe

Petite pratique avec les filtres linéaires



Originale

0	0	0
0	2	0
0	0	0

-

$\frac{1}{9}$

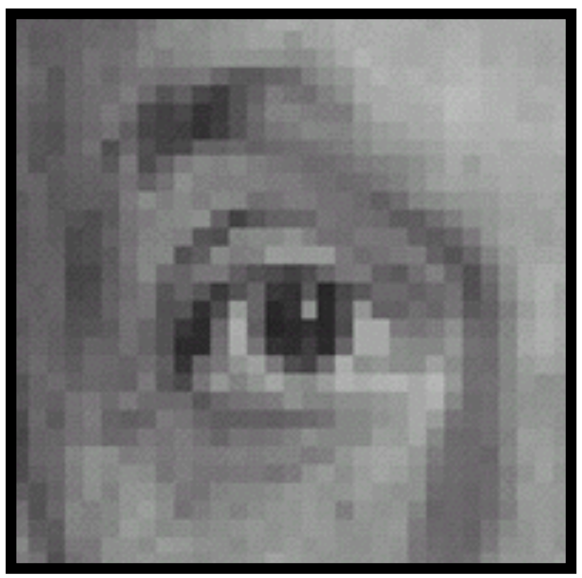
1	1	1
1	1	1
1	1	1

?

(Notez que la somme est égale à 1)

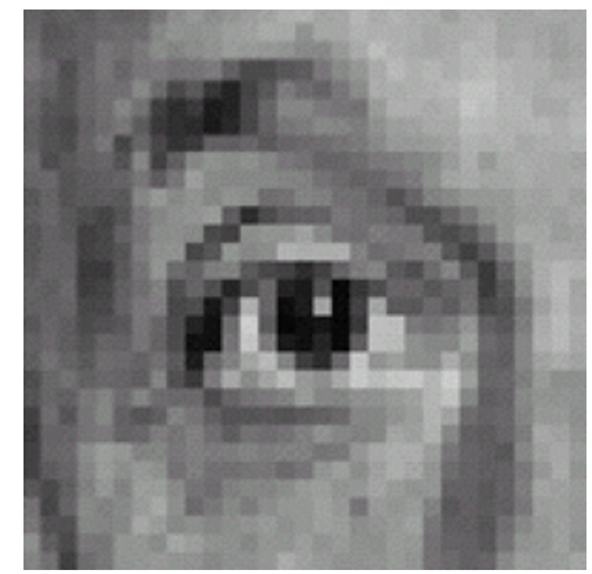
Source: D. Lowe

Petite pratique avec les filtres linéaires



Originale

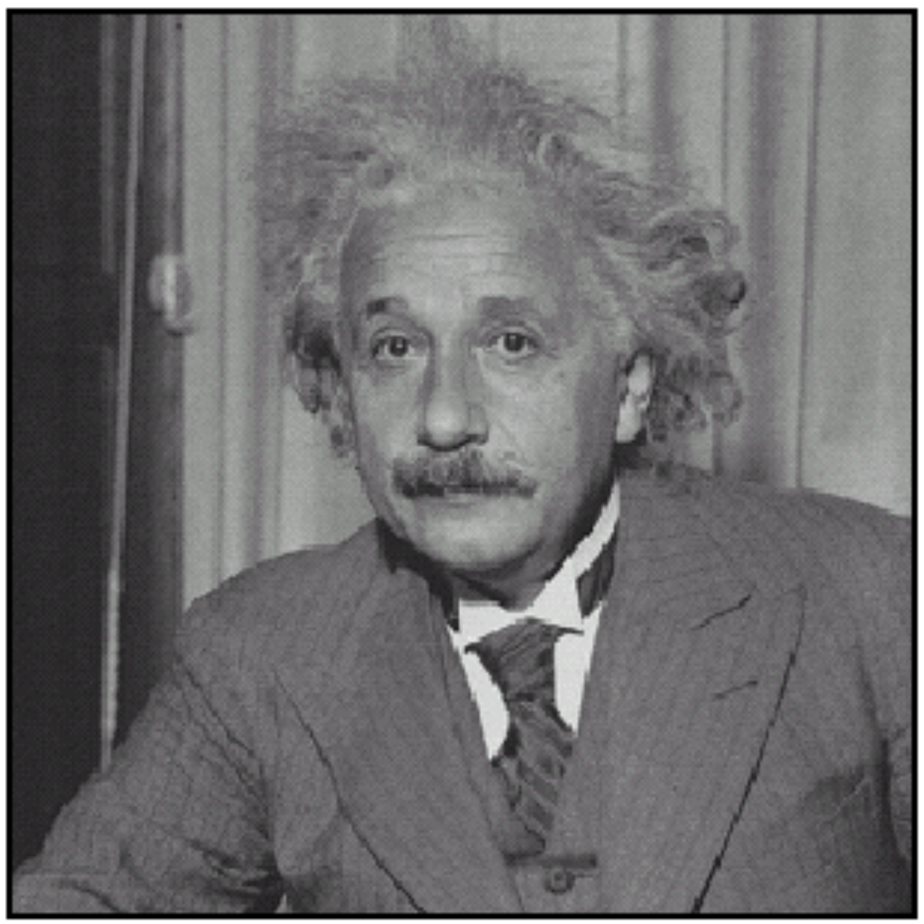
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

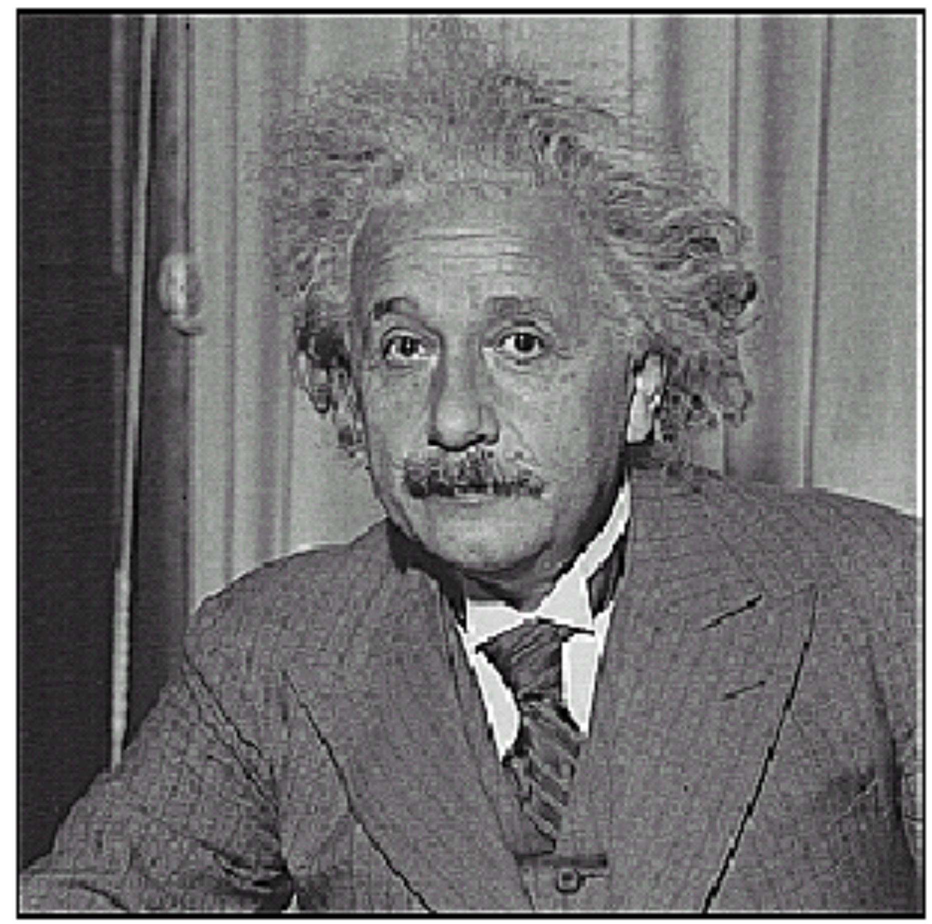


Accentue les différences avec la moyenne

Source: D. Lowe

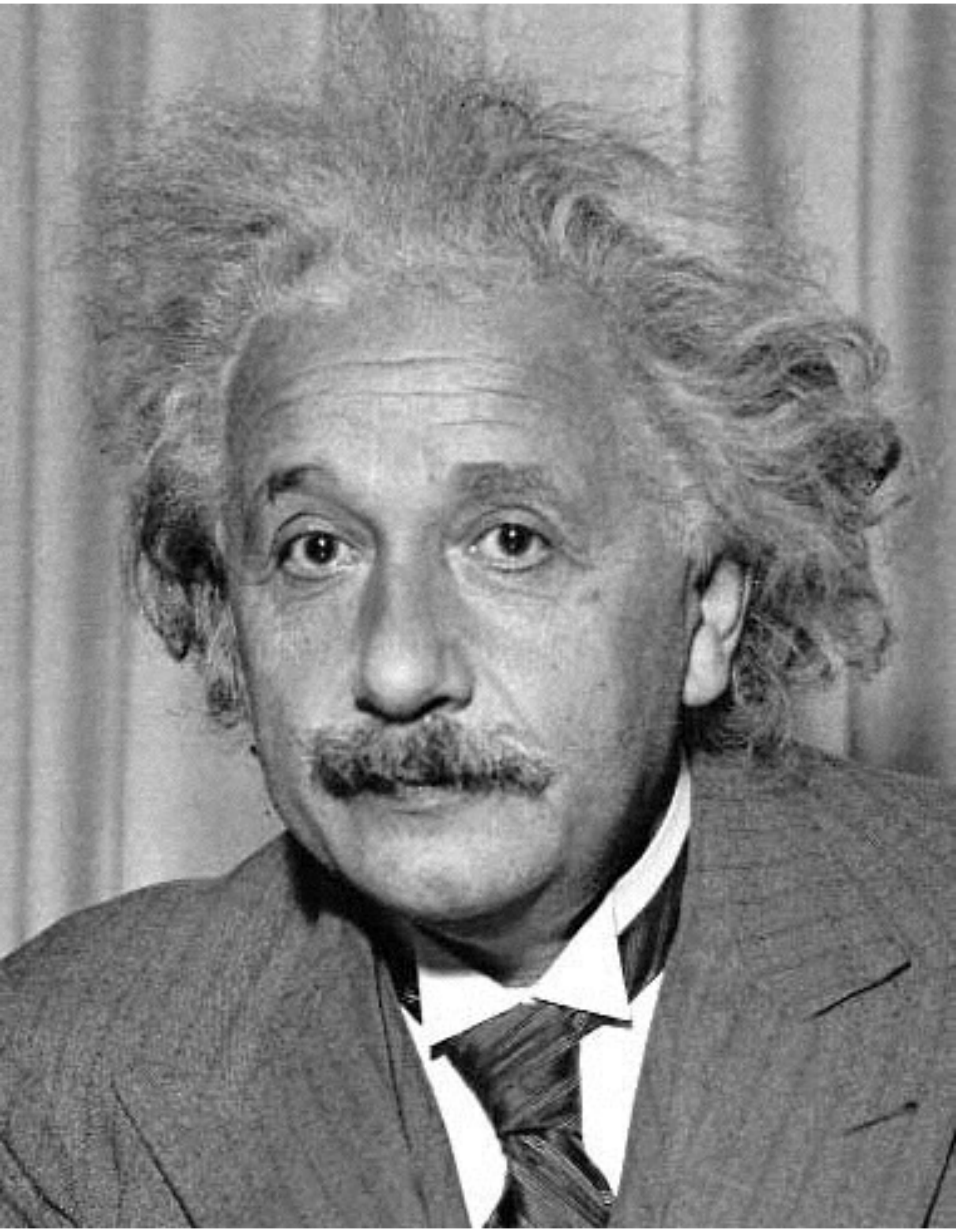
Accentuation “sharpening”





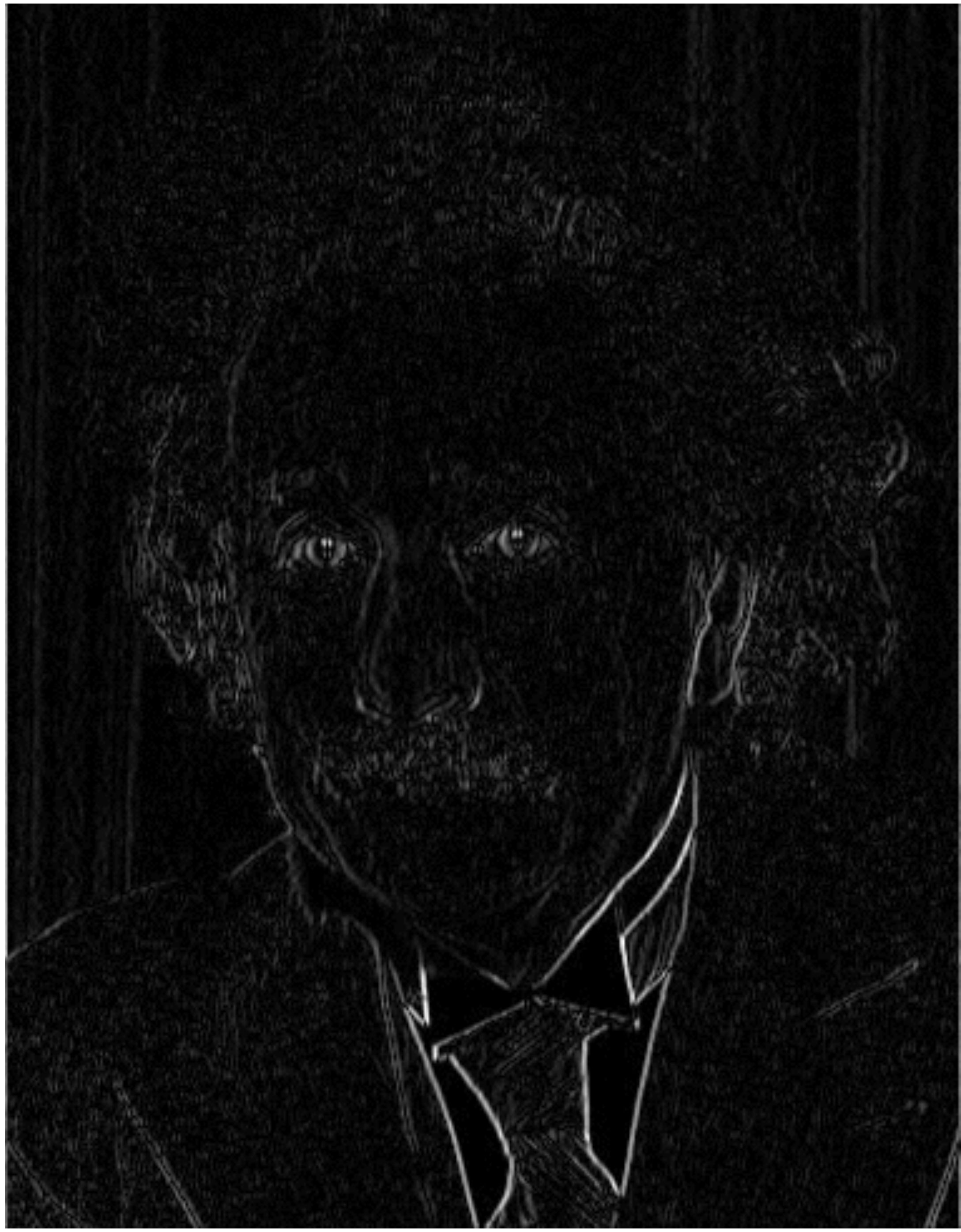
Source: D. Lowe

Autres filtres



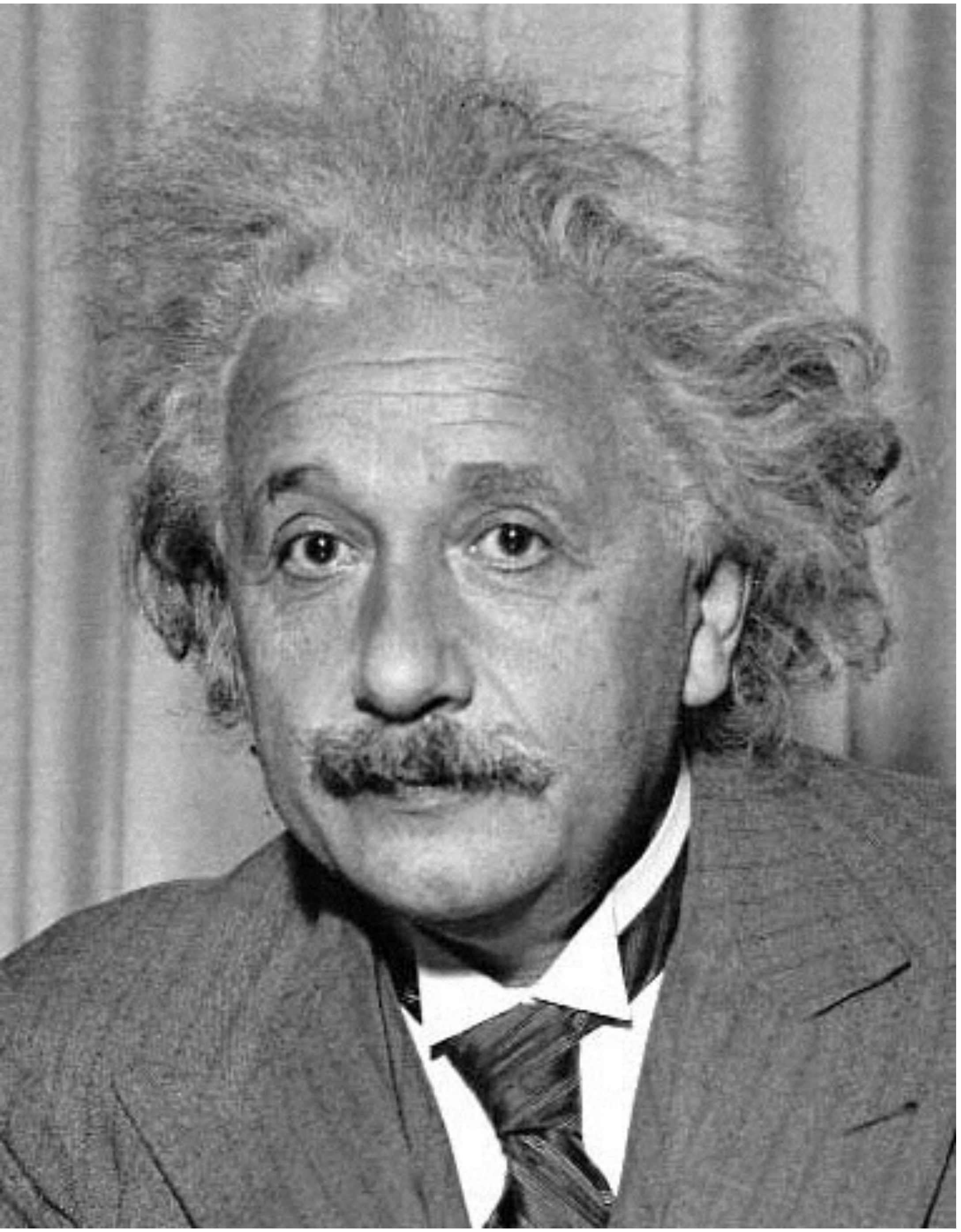
1	0	-1
2	0	-2
1	0	-1

Sobel



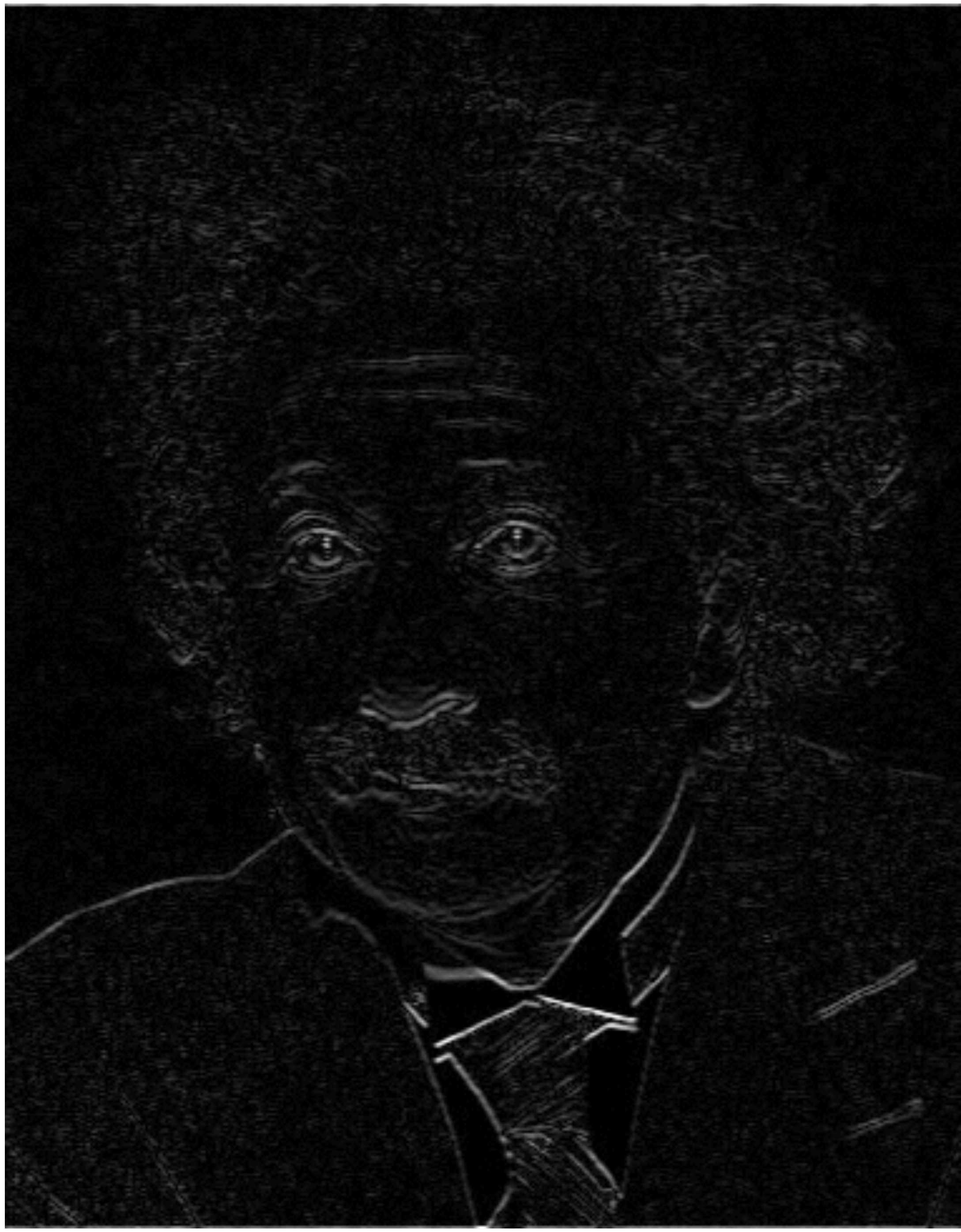
Arêtes verticales
(valeur absolue)

Autres filtres



1	2	1
0	0	0
-1	-2	-1

Sobel



Arêtes horizontales
(valeur absolue)

Démo!

Propriété des filtres linéaires

- Linéarité (!):
  - filtre(f1 + f2) = filtre(f1) + filtre(f2)
- Invariance aux déplacement: ne dépend pas de l'emplacement du pixel
  - filtre(shift(f)) = shift(filtre(f))

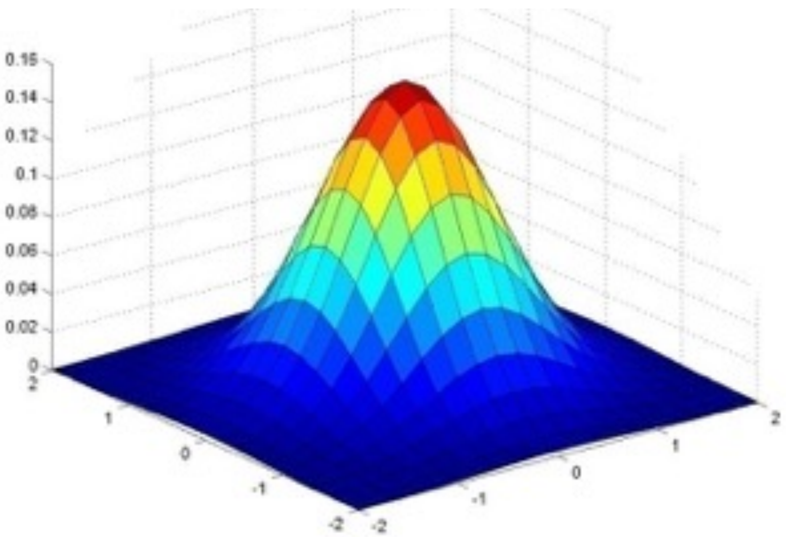
Source: S. Lazebnik

Propriétés des filtres linéaires

- Commutatif: $a * b = b * a$
  - On peut donc traiter l'image comme un filtre
- Associatif: $a * (b * c) = (a * b) * c$
  - Souvent, on applique plusieurs filtres: $(((a * b1) * b2) * b3)$
  - Équivalent à filtrer l'image avec le "filtre du filtre": $a * (b1 * b2 * b3)$
- Distribution: $a * (b + c) = (a * b) + (a * c)$
- Factoriser les scalaires: $ka * b = a * kb = k (a * b)$
- Identité $e = [0, 0, 1, 0, 0]$, $a * e = a$

Source: S. Lazebnik

Filtre important: gaussien



0.003	0.013	0.022	0.013	0.003
0.013	0.059	0.097	0.059	0.013
0.022	0.097	0.159	0.097	0.022
0.013	0.059	0.097	0.059	0.013
0.003	0.013	0.022	0.013	0.003

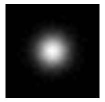
5 x 5, σ = 1

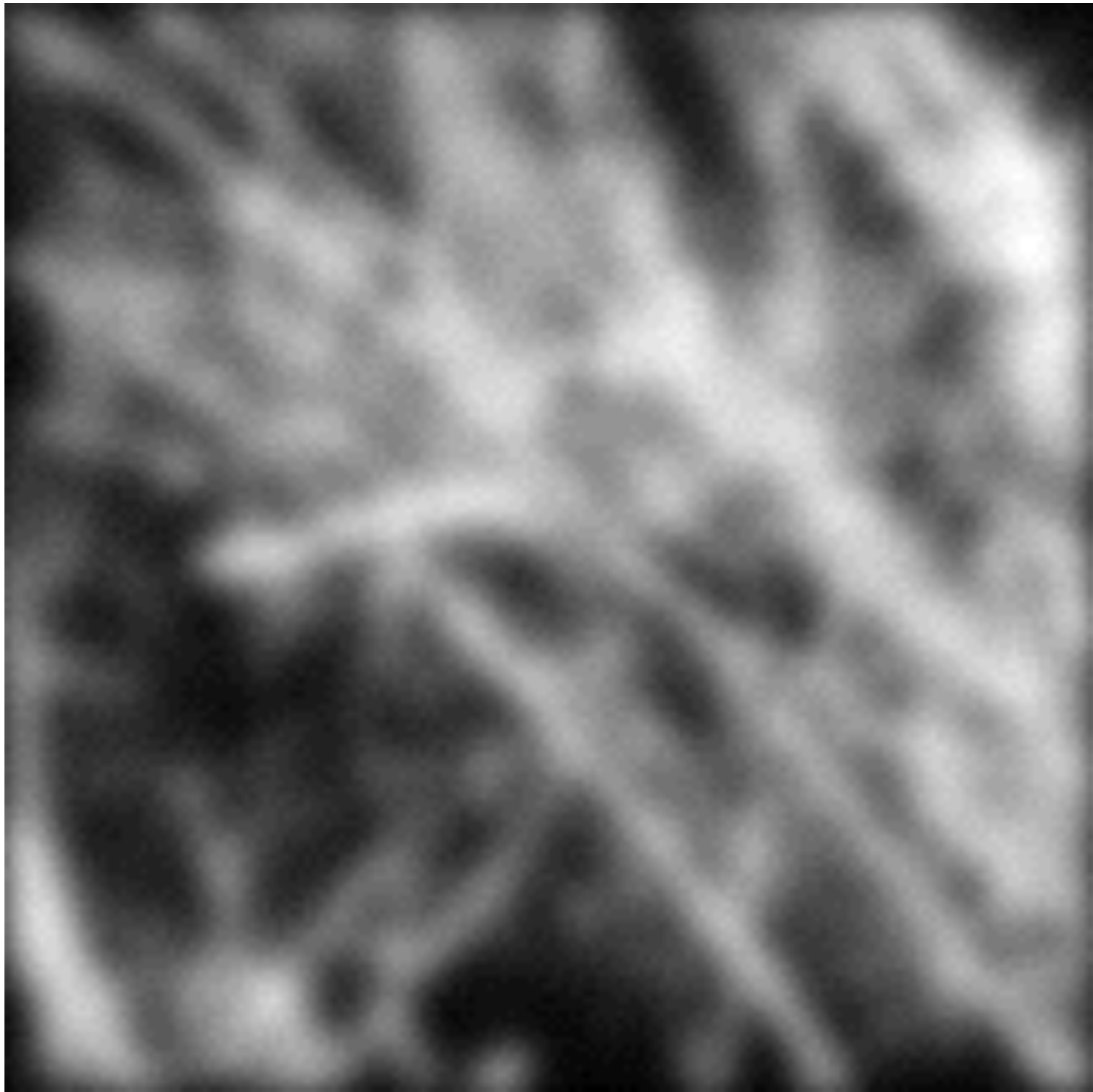
$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

- Pondère les contributions des voisins en fonction de leur distance

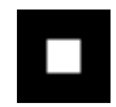
Slide credit: Christopher Rasmussen

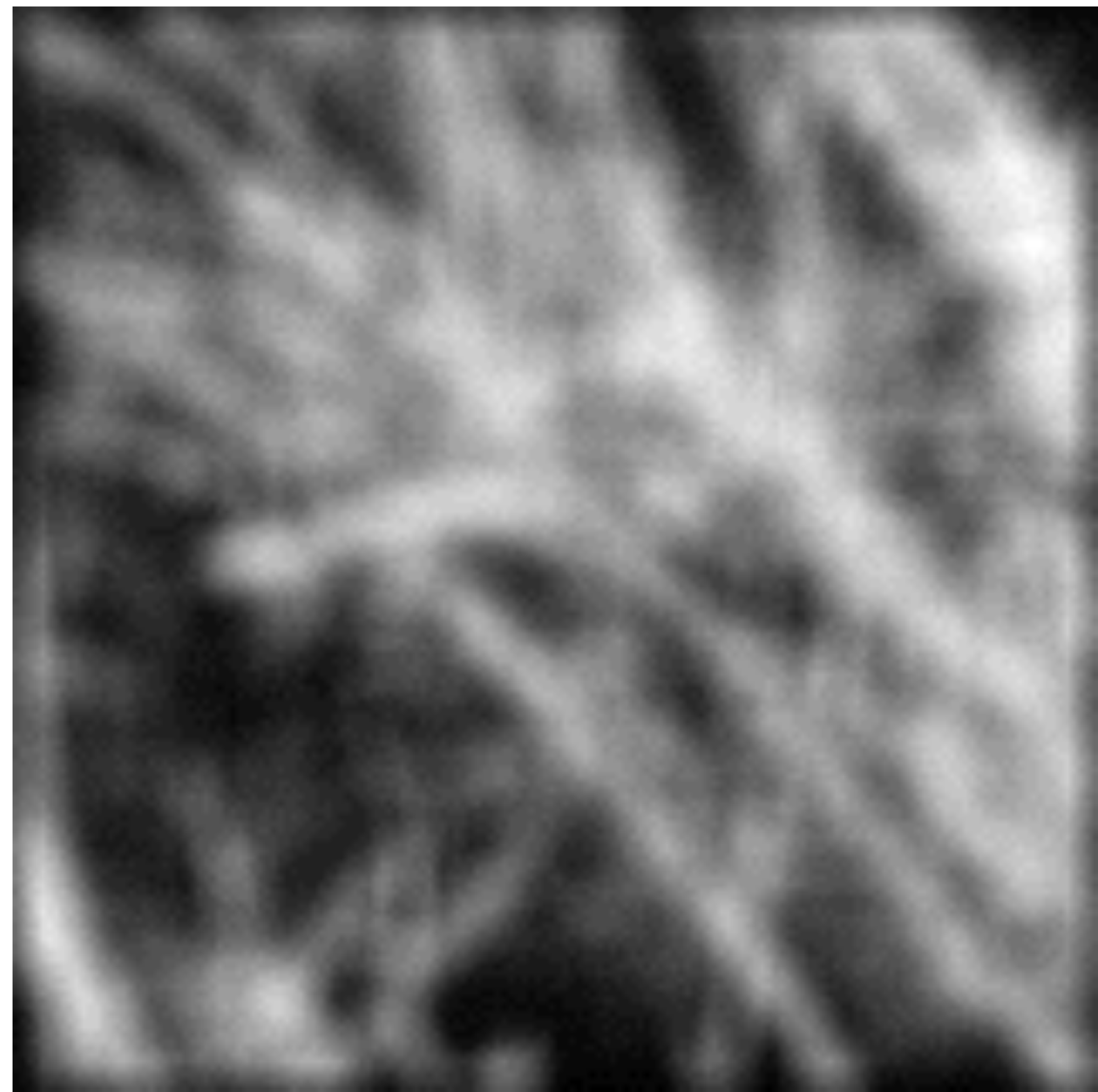
Atténuer avec le filtre gaussien





Atténuer avec le filtre “en boîte”





Filtres gaussiens

- Retire les hautes fréquences de l'image (filtre passe-bas)
    - Les images deviennent plus lisses
- Filter un filtre gaussien avec un autre filtre gaussien?
    - Le résultat est aussi gaussien
        - Si les deux filtres ont une variance de sigma, c'est équivalent à un filtre de variance sigma*sqrt(2)
- Séparable
    - filtre gaussien 2D = produit de deux filtres gaussiens 1D

Source: K. Grauman

Séparabilité du filtre gaussien

$$
\begin{aligned}
G_\sigma(x, y) &= \frac{1}{2\pi\sigma^2} \exp^{-\frac{x^2 + y^2}{2\sigma^2}} \\
&= \left(\frac{1}{\sqrt{2\pi}\sigma} \exp^{-\frac{x^2}{2\sigma^2}} \right) \left(\frac{1}{\sqrt{2\pi}\sigma} \exp^{-\frac{y^2}{2\sigma^2}} \right)
\end{aligned}
$$

Pourquoi c'est important?

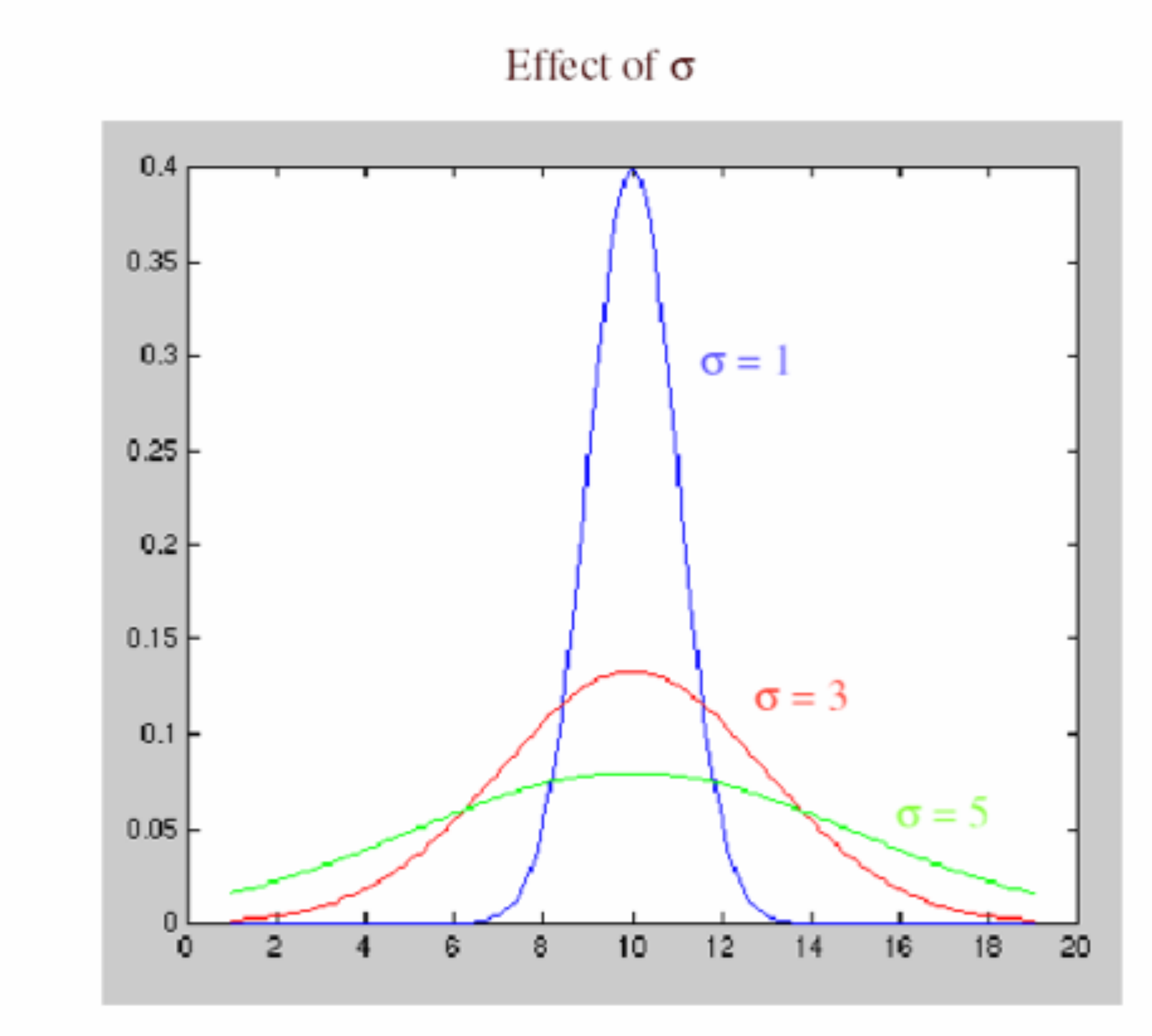
Source: D. Lowe

Démonstration

(separable.m)

Considérations pratiques

- Quelle devrait être la taille du filtre?
- Les valeurs aux extrémités devraient être proche de 0
- Règle empirique: la demie-taille devrait être 3σ



Considérations pratiques

- Bordure de l'image?
  - le filtre dépasse l'image!
  - extrapolation
    - 0
    - "enrouler"
    - répéter
    - réflexion

Source: S. Marschner

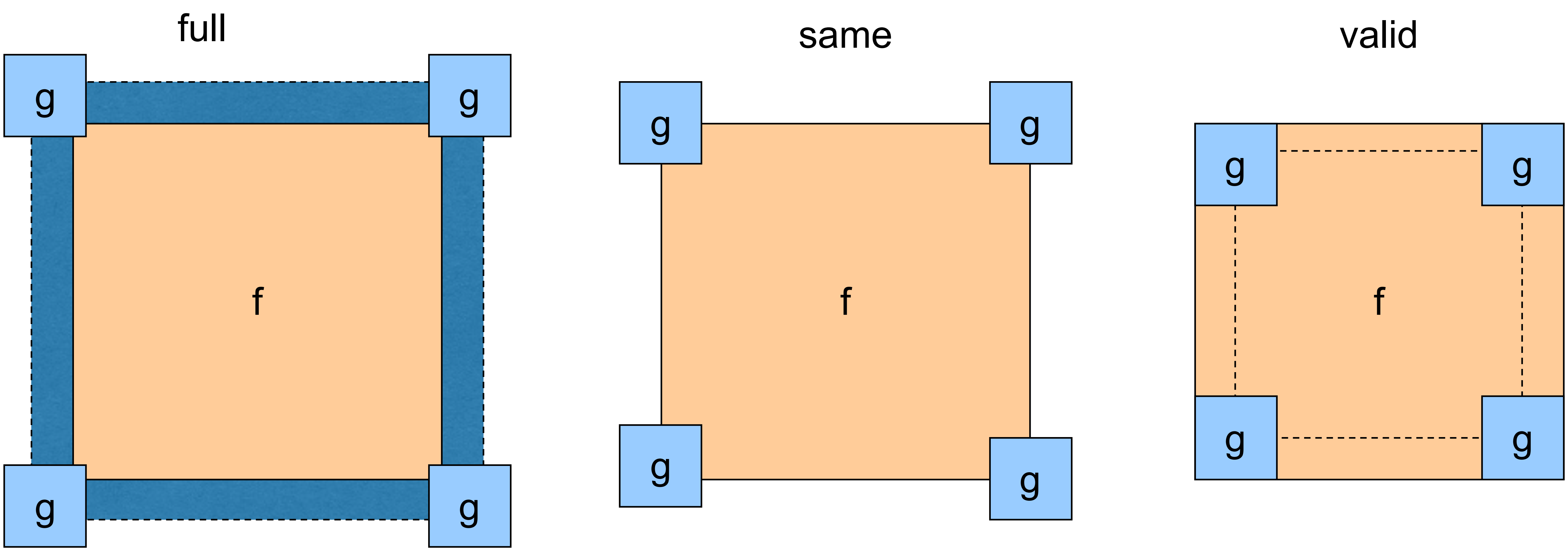
Considérations pratiques

- (MATLAB):
  - 0: imfilter(f, g, 0)
  - “enrouler”: imfilter(f, g, ‘circular’)
  - répéter: imfilter(f, g, ‘replicate’)
  - réflexion: imfilter(f, g, ‘symmetric’)

Source: S. Marschner

Considérations pratiques

- Quelle est la taille de l'image résultante?
- MATLAB: filter2(g, f, shape)
  - shape = 'full': size(f) + size(g)
  - shape = 'same': size(f)
  - shape = 'valid': size(f)-size(g)



Source: S. Lazebnik

Questions de révision

- Un filtre 3x3 qui retourne un nombre positif si la valeur moyenne des 4-voisins est plus petite que celle du pixel central, et négatif sinon.
- Un filtre qui calcule le gradient dans la direction horizontale:
  - gradx(y,x) = im(y,x+1) - im(y,x) pour tout x, y

Questions de révision

Remplir les trous:

Filtrage

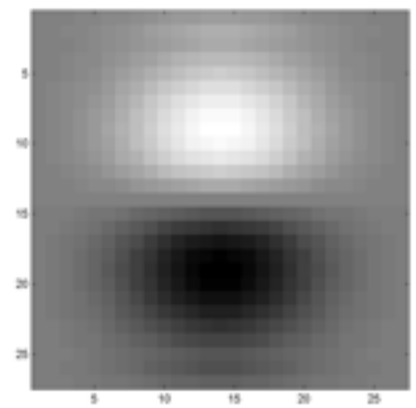
a) _ = D * B

b) A = _ * _

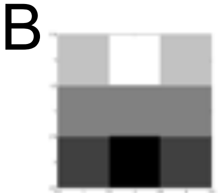
c) F = D * _

d) _ = D * D

A

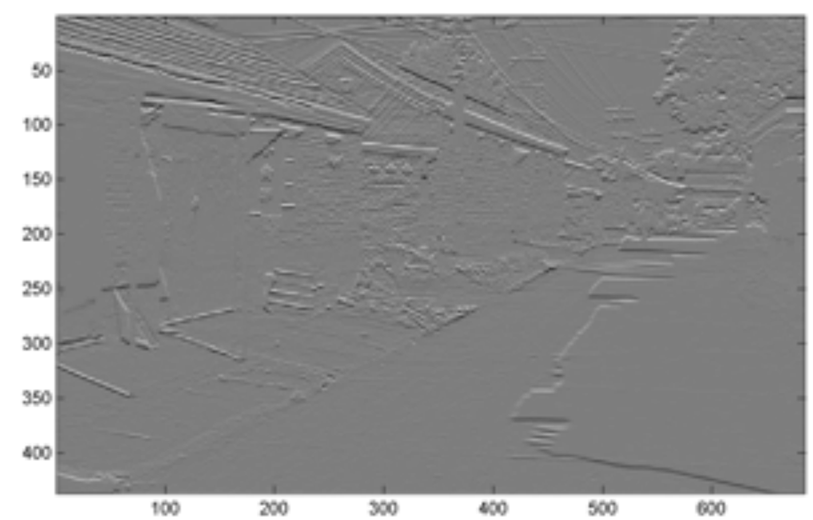


B

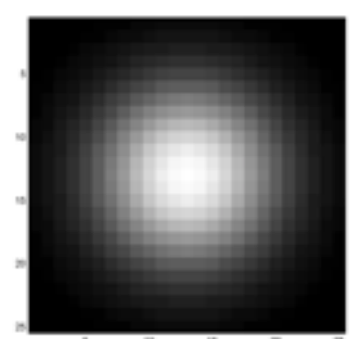


E

G



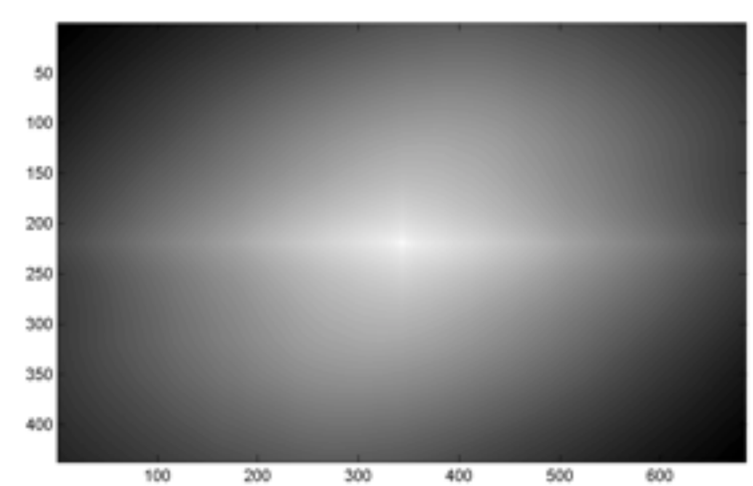
C



F

H

I



D