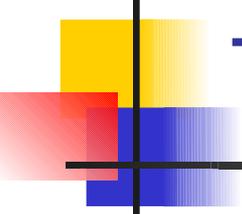


# Vision par ordinateur

---

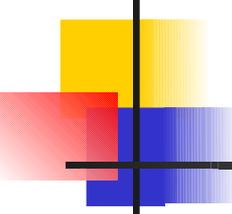
**Texture**



# Texture

---

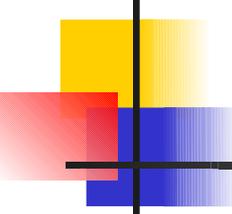
- Les textures sont
  - une information intéressante à analyser
  - mais plus difficile à extraire
- Jusqu'à présent, on définissait
  - Une **région** comme une zone homogène
  - Un **contour** comme une variation d'intensité
- Une **texture** peut être définie
  - comme une région avec des variations d'intensité
  - comme une organisation spatiale des pixels
  - ...



# Exemples de textures

---

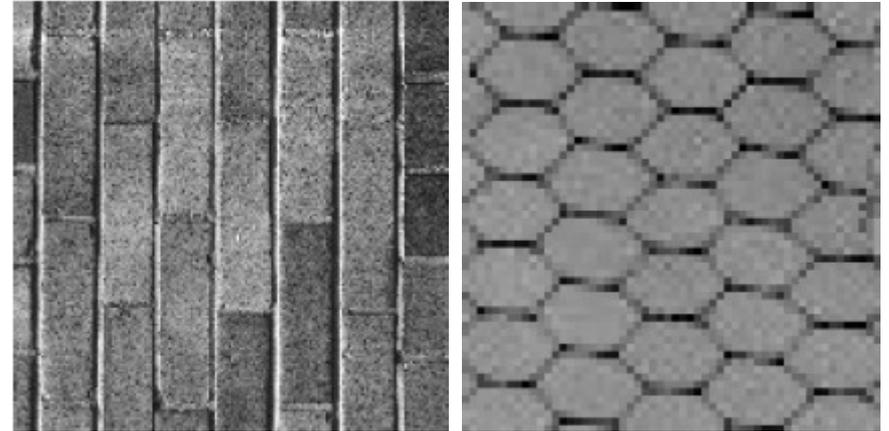




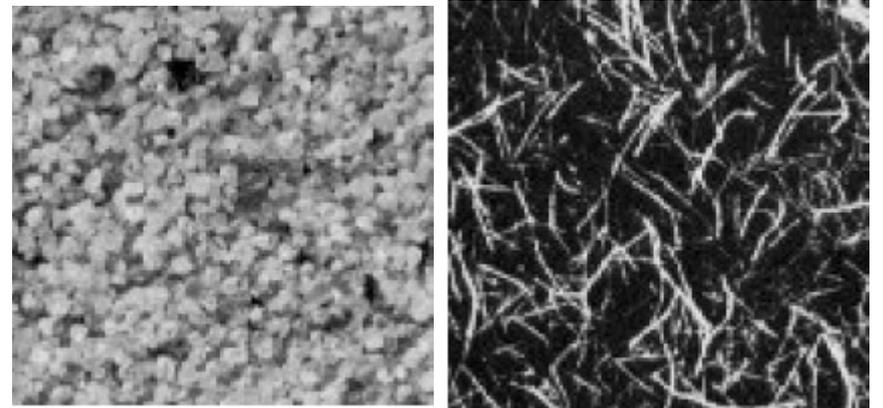
# Motifs des textures

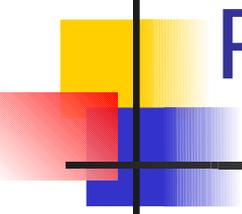
---

Une texture peut être **périodique**  
(répétition d'un motif  
de base) ...



...ou **non-périodique**  
(pas de motif de base,  
plus désordonnée)

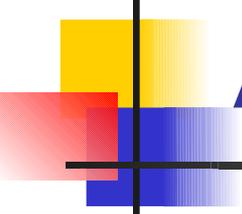




# Plusieurs sous-problèmes

---

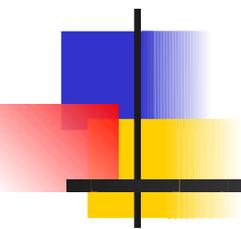
- **Segmentation de texture**
  - Diviser une image en plusieurs régions ou chaque région représente une même texture
- **Synthèse de texture**
  - A l'aide de petites images de texture, construire de plus grandes images en extrapolant ces textures
- **Shape from Texture**
  - Reconstruction de la forme 3D d'un objet en analysant sa texture



# Analyse de la texture

---

- Il existe plusieurs méthodes pour analyser la texture :
  - Statistiques du premier ordre
    - Statistiques sur l'histogramme
  - Matrices de co-occurrences
    - Recherche de motifs
  - Analyse fréquentielle
    - Filtres de Gabor
- Le plus difficile est de trouver une bonne représentation (paramètres) pour chaque texture

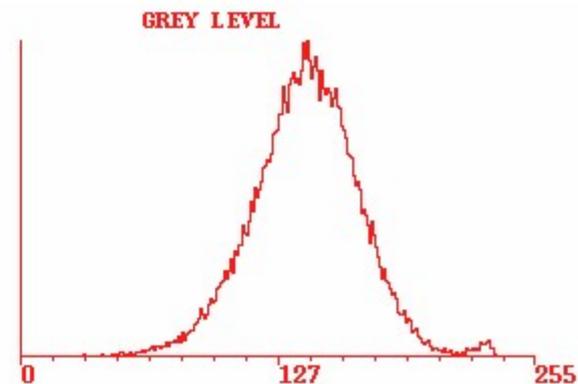
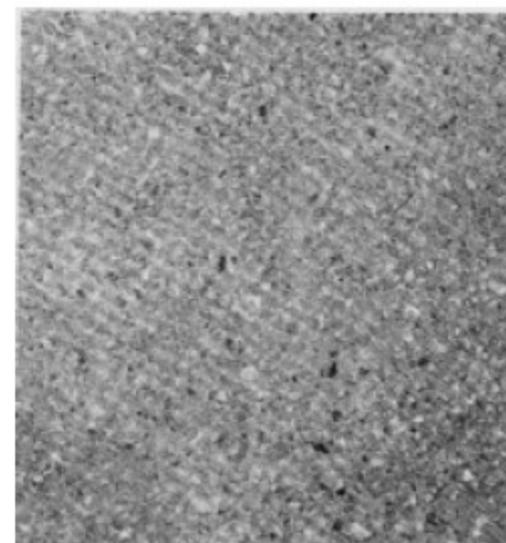
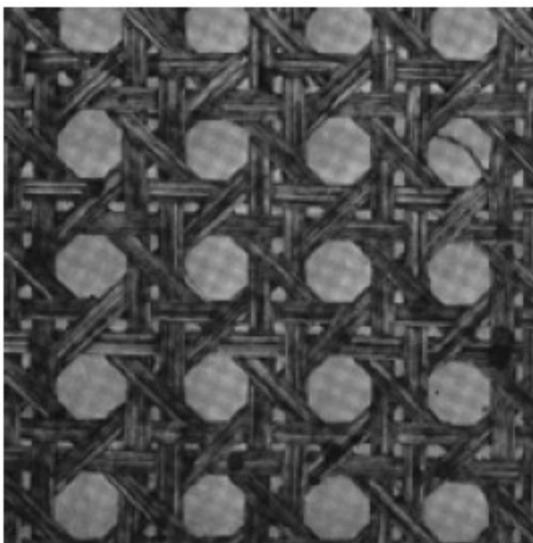


# Analyse de la texture

---

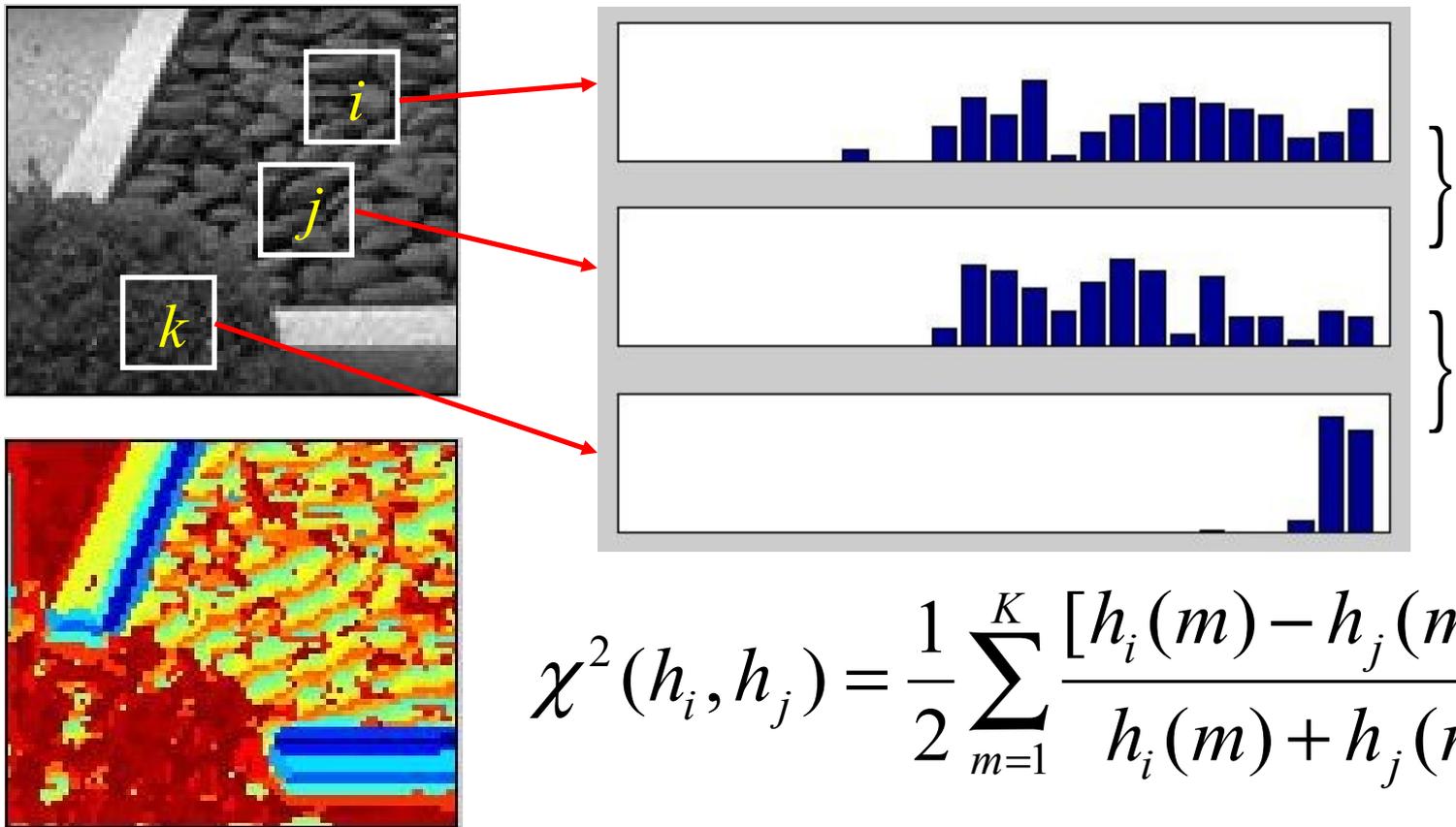
Statistiques du premier ordre

# Histogrammes et textures

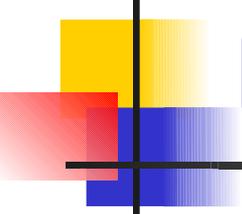


# Distance d'histogrammes

## *Distance du $\chi^2$ entre histogrammes de textures*



$$\chi^2(h_i, h_j) = \frac{1}{2} \sum_{m=1}^K \frac{[h_i(m) - h_j(m)]^2}{h_i(m) + h_j(m)}$$



# Paramètres des textures

---

- On peut calculer plusieurs paramètres d'une texture à partir de l'image ou de son histogramme :

$$MOYENNE = \sum_y \sum_x g(x, y)$$

$$VARIANCE = \frac{1}{N} \sum_y \sum_x (g(x, y) - MOYENNE)^2$$

$$SKEWNESS = \frac{1}{N} \sum_y \sum_x (g(x, y) - MOYENNE)^3$$

$$KURTOSIS = \frac{1}{N} \sum_y \sum_x (g(x, y) - MOYENNE)^4$$

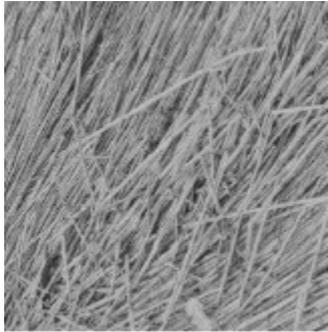
**où**

**$g(x,y)$**  : niveaux de gris des pixels

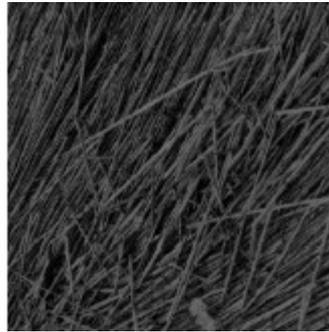
Ce sont les moments d'ordre 1,2,3,4,... de l'image.

D'autres paramètres existent (uniformité, entropie, ...)

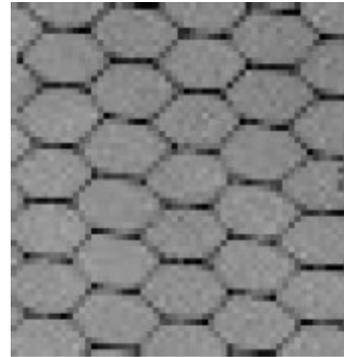
# Paramètres des textures



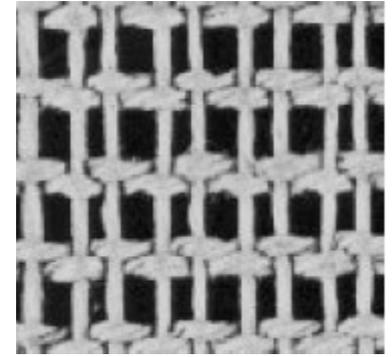
*moyenne=137*



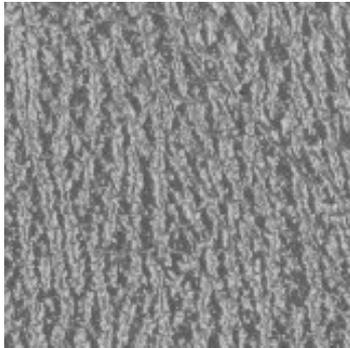
*moyenne=50*



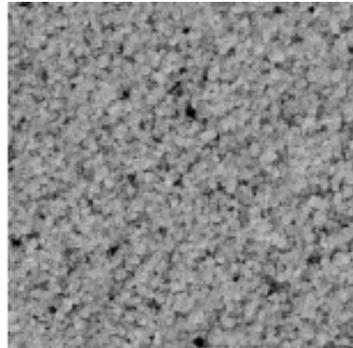
*variance=30*



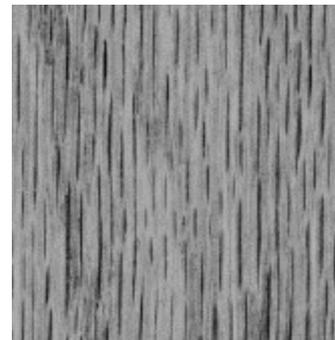
*variance=71*



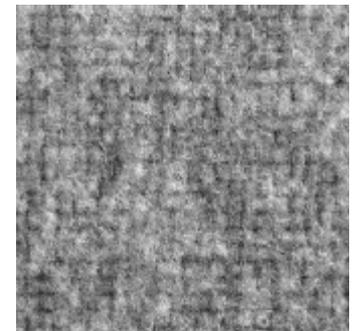
*skew=0.25*



*skew=-1.15*

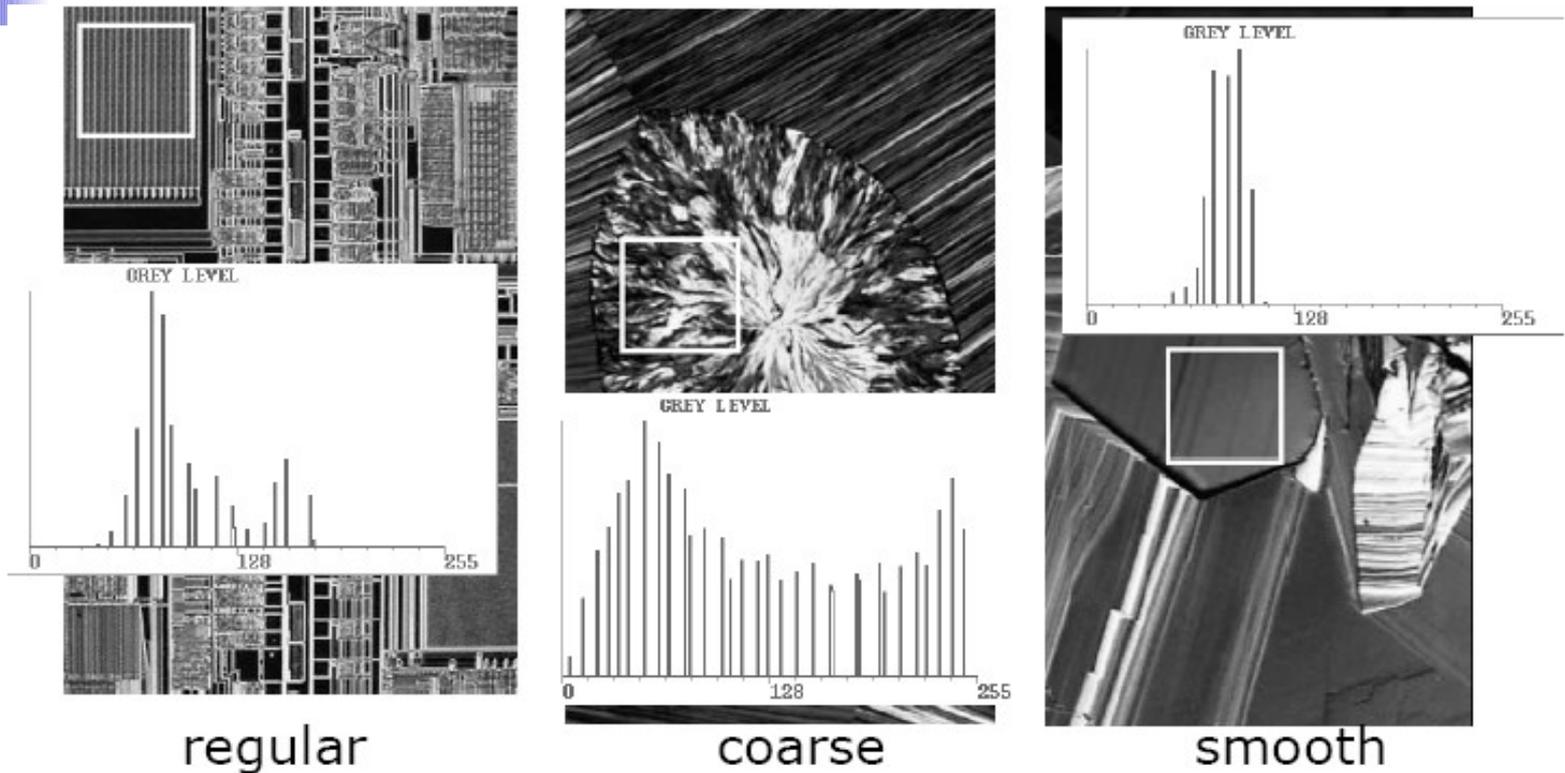


*kurt=2.31*

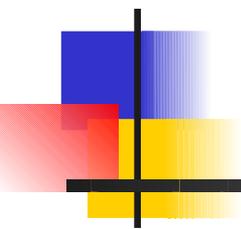


*kurt=-0.09*

# Caractérisation des textures



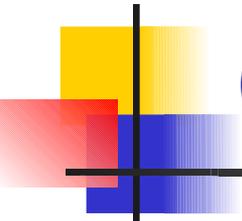
texture	$m_1$	$\sigma$	$R_{\text{norm}}$	$\mu_3$	U	e
regular	99.72	33.73	0.017	0.750	0.013	6.674
coarse	143.56	74.63	0.079	-0.151	0.005	7.783
smooth	82.64	11.79	0.002	-0.105	0.026	5.434



# Analyse de la texture

---

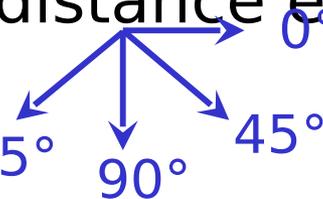
Matrices de co-occurrences



## (2) Matrices de co-occurrences

---

- L'idée de cette méthode est d'identifier les **répétitions de niveaux de gris** selon une **distance** et une **direction**
  - *Matrices de co-occurrence (Haralick)*
- **Matrice de taille  $N_g \times N_g$** 
  - $N_g$  étant le nombre de niveaux de gris de l'image (256x256)
  - On réduit souvent à des tailles 8x8, 16x16 ou 32x32
- Plusieurs matrices, pour chaque distance et direction
  - **Distance** : 1, 2, 3 (, 4, ...)
  - **Direction** : 0°, 45°, 90°, 135° (, ..135°
- Temps de calcul des matrices est assez long



# Matrices de co-occurrences

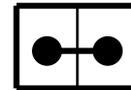
- Exemple de calcul des matrices :

1	4	4	3
4	2	3	2
1	2	1	4
1	2	2	3

*Image*

	1	2	3	4
1	?	?	?	?
2	?	?	?	?
3	?	?	?	?
4	?	?	?	?

*Matrice pour  
distance=1 et  
direction=0°*



*On parcourt l'image et pour chaque couple de pixels formé avec la distance et la direction données, on incrémente la matrice des co-occurrences de 1*

# Matrices de co-occurrences

- Exemple de calcul des matrices :



<b>1</b>	<b>4</b>	4	3
4	2	3	2
1	2	1	4
1	2	2	3

*Image*

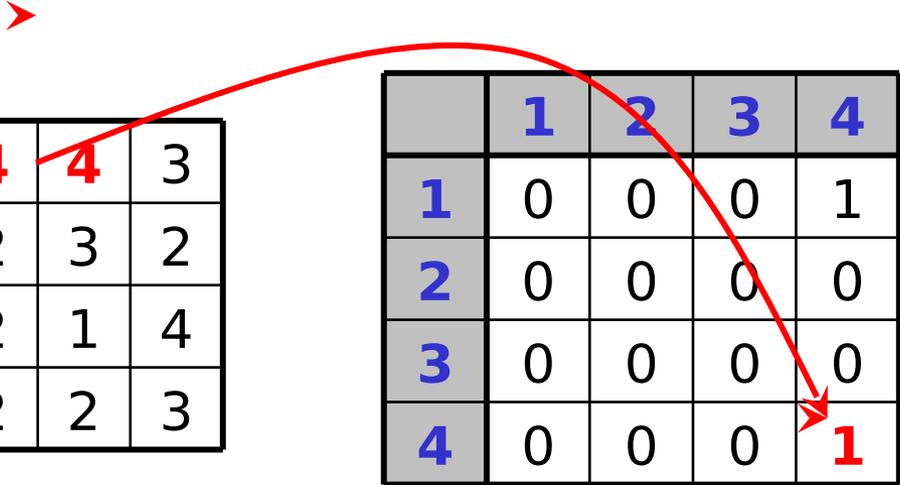
	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>1</b>	0	0	0	<b>1</b>
<b>2</b>	0	0	0	0
<b>3</b>	0	0	0	0
<b>4</b>	0	0	0	0

*Matrice pour distance=1  
et direction=0°*

*Couple de pixels voisins (1,4)*

# Matrices de co-occurrences

- Exemple de calcul des matrices :



1	<b>4</b>	<b>4</b>	3
4	2	3	2
1	2	1	4
1	2	2	3

*Image*

	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>1</b>	0	0	0	1
<b>2</b>	0	0	0	0
<b>3</b>	0	0	0	0
<b>4</b>	0	0	0	<b>1</b>

*Matrice pour distance=1  
et direction=0°*

*Couple de pixels voisins (4,4)*

# Matrices de co-occurences

- Exemple de calcul des matrices :

<b>1</b>	<b>4</b>	<b>4</b>	<b>3</b>
<b>4</b>	<b>2</b>	<b>3</b>	<b>2</b>
1	2	1	4
1	2	2	3

*Image*

	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>1</b>	0	0	0	<b>1</b>
<b>2</b>	0	0	<b>1</b>	0
<b>3</b>	0	<b>1</b>	0	0
<b>4</b>	0	<b>1</b>	<b>1</b>	<b>1</b>

*Matrice pour distance=1  
et direction=0°*

*etc. (après 2 lignes)*

# Matrices de co-occurrences

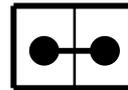
- Exemple de calcul des matrices (final) :

1	4	4	3
4	2	3	2
1	2	1	4
1	2	2	3

*Image*

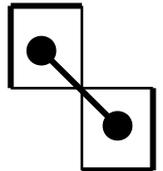
	1	2	3	4
1	0	2	0	2
2	1	1	2	0
3	0	1	0	0
4	0	1	1	1

*Matrice pour  
distance=1 et  
direction=0°*

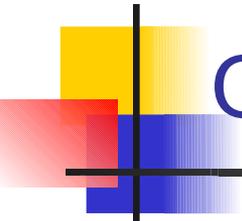


	1	2	3	4
1	0	2	1	0
2	1	1	0	0
3	0	0	0	1
4	0	2	1	0

*Matrice pour  
distance=1 et  
direction=45°*



*...et ainsi de suite pour chaque matrice (plusieurs au total)*



# Calculs des paramètres des matrices

- Pour chaque matrice, on peut calculer **quatorze paramètres** caractérisant la texture, dont les plus importants :

$$MOYENNE = \sum_j \sum_i p(i, j)$$

$$VARIANCE_i = \sum_j \sum_j (i - MOYENNE)^2 p(i, j)$$

$$ENERGIE = \sum_j \sum_i (p(i, j))^2$$

$$INERTIE = \sum_j \sum_i (i - j)^2 p(i, j)$$

$$ENTROPIE = - \sum_j \sum_i p(i, j) \log(p(i, j))$$

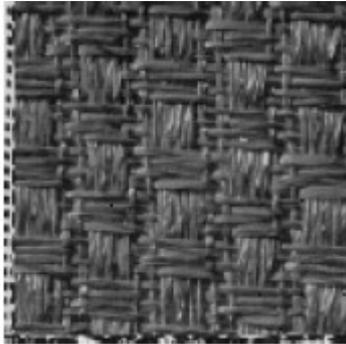
$$MOMENT DIFFERENTIEL INVERSE = \sum_i \sum_j \frac{1}{1 + (i - j)^2} p(i, j)$$

**où**

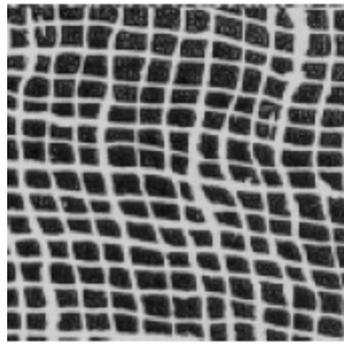
**(i,j)** : coordonnées dans la matrice

**p(i,j)** : valeurs normalisées de la matrice

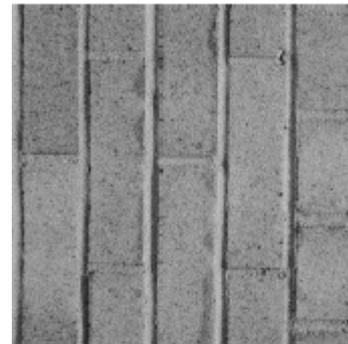
# Paramètres de la matrice



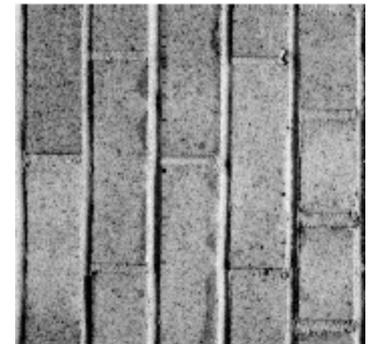
*contraste=120*



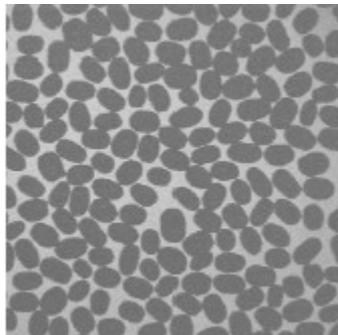
*contraste=626*



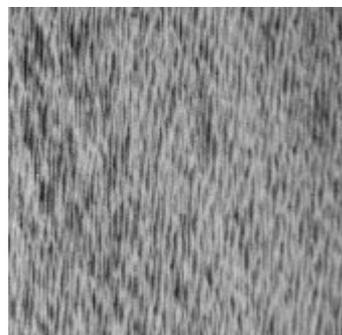
*contraste=42*



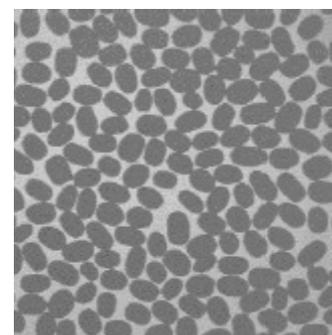
*contraste=142*



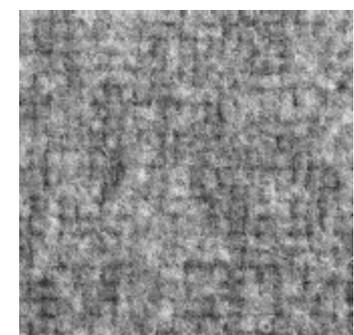
*entropie=6.7*



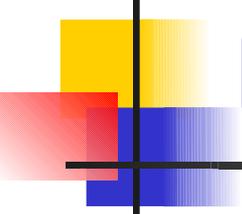
*entropie=4.7*



*mdi=0.03*



*mdi=0.002*

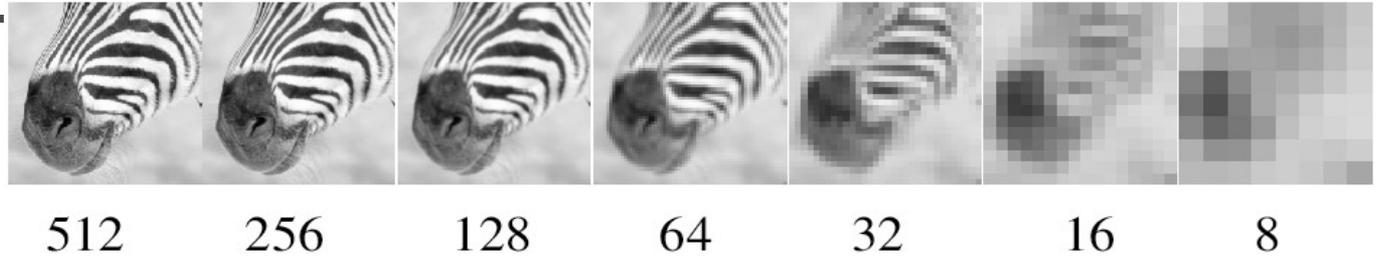


# Détection des textures

---

- Avec les matrices de co-occurrences, on espère identifier une texture particulière (selon une distance et une direction à trouver)
- Problème du calcul sur l'image entière ou sur des zones de l'image (découpage en carrés de l'image)
- Il existe plusieurs variantes de cette méthode basée sur le calcul de différences de niveaux de gris

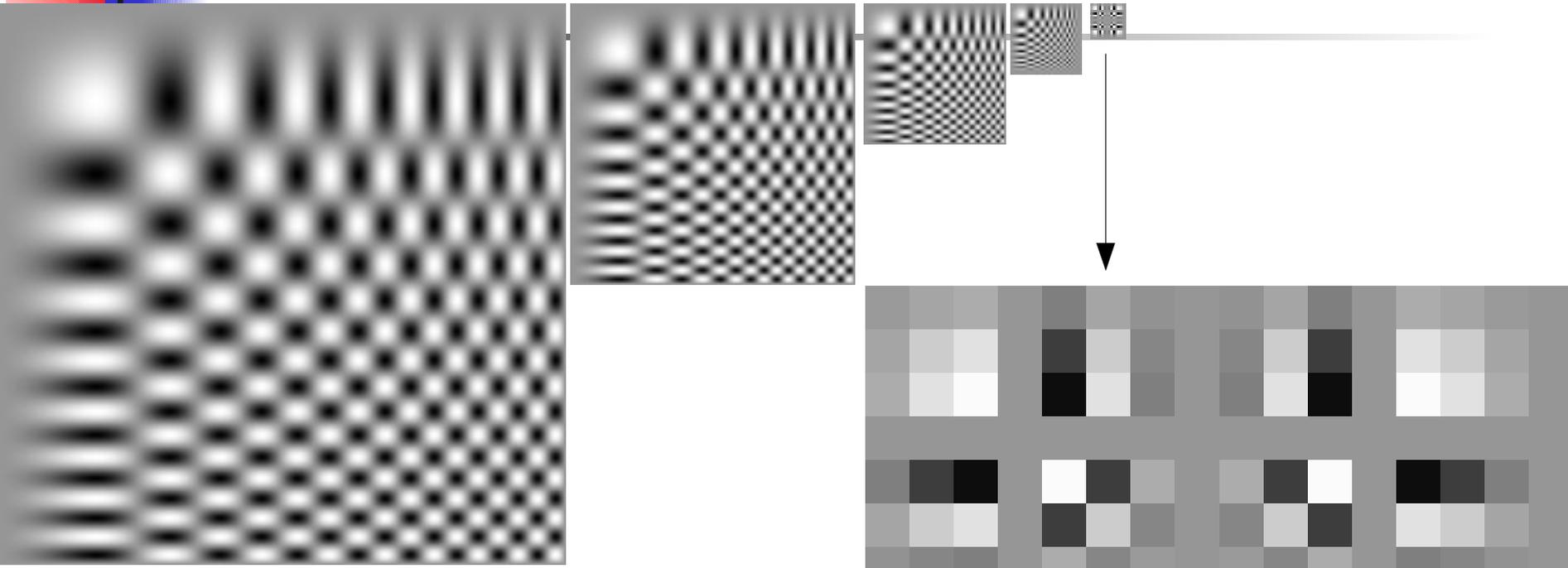
# Problème d'échelle pour les textures



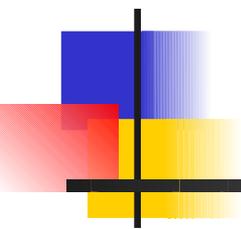
*Pour bien identifier les textures, on a souvent besoin de travailler à différentes échelles...*



# Alliassage des textures



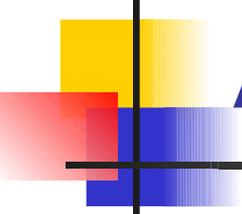
*Lorsqu'on réduit trop l'image de départ, on obtient de mauvaises représentation des textures : phénomène d'alliassage (que nous étudierons en synthèse d'images)*



# Analyse de la texture

---

Analyse fréquentielle



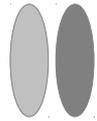
# Analyse fréquentielle

---

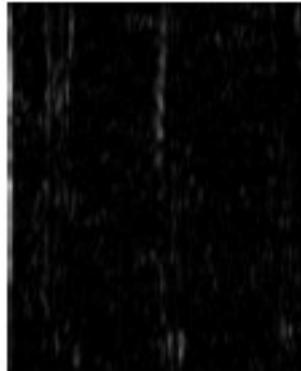
- Une texture est une répétition d'éléments avec une certaine **fréquence**
- Filtrage de l'image en utilisant des méthodes spectrales
  - Fourier
  - **Filtres de Gabor**
  - ...
- Pour rappel sur la Transformée de Fourier (FFT)
  - *(re)voir le cours de Traitement d'images*

# Idée générale du filtrage fréquentiel

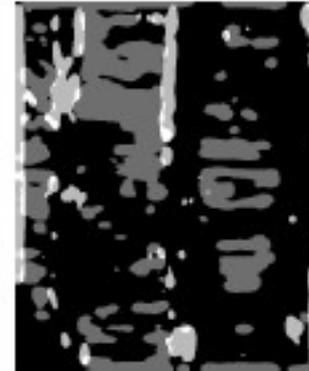
Réponses au carré



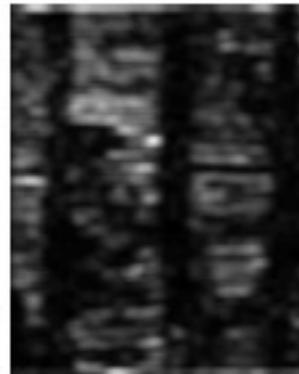
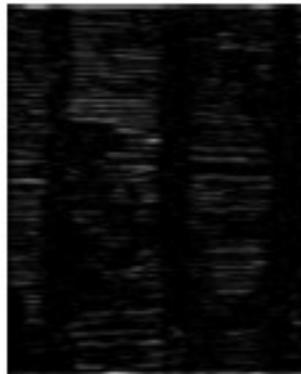
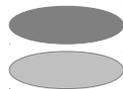
vertical



classification



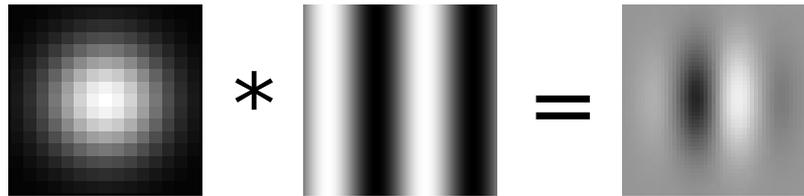
horizontal



Lissage

# Filtres de Gabor

- Les filtres de Gabor sont le produit de Gaussiennes par des sinus ou des cosinus
  - Sinus / cosinus : analyse en fréquences
  - Gaussienne : différents niveaux de lissage (échelles)



Gaussienne \* Sinus/Cosinus = Gabor

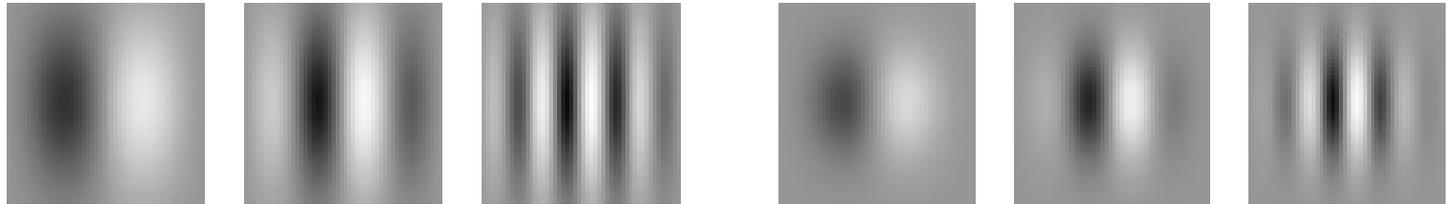
$$G_s(x, y) = \cos(\omega_x x + \omega_y y) e^{-\frac{x^2 + y^2}{2\sigma^2}}$$

$$G_a(x, y) = \sin(\omega_x x + \omega_y y) e^{-\frac{x^2 + y^2}{2\sigma^2}}$$

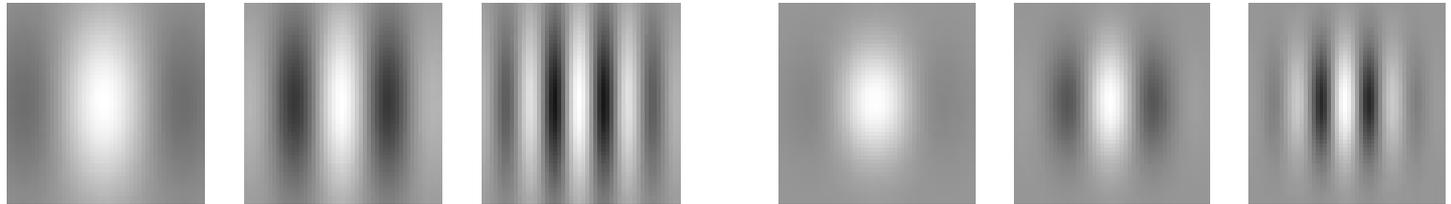
# Filtres de Gabor

- Les filtres de Gabor analysent la structure
  - à différentes échelles (fréquences)
  - à différentes orientations (angles)

Filtres  
anti-  
symétriques  
(impairs -  
sinus)



Filtres  
symétriques  
(pairs -  
cosinus)



Grande échelle

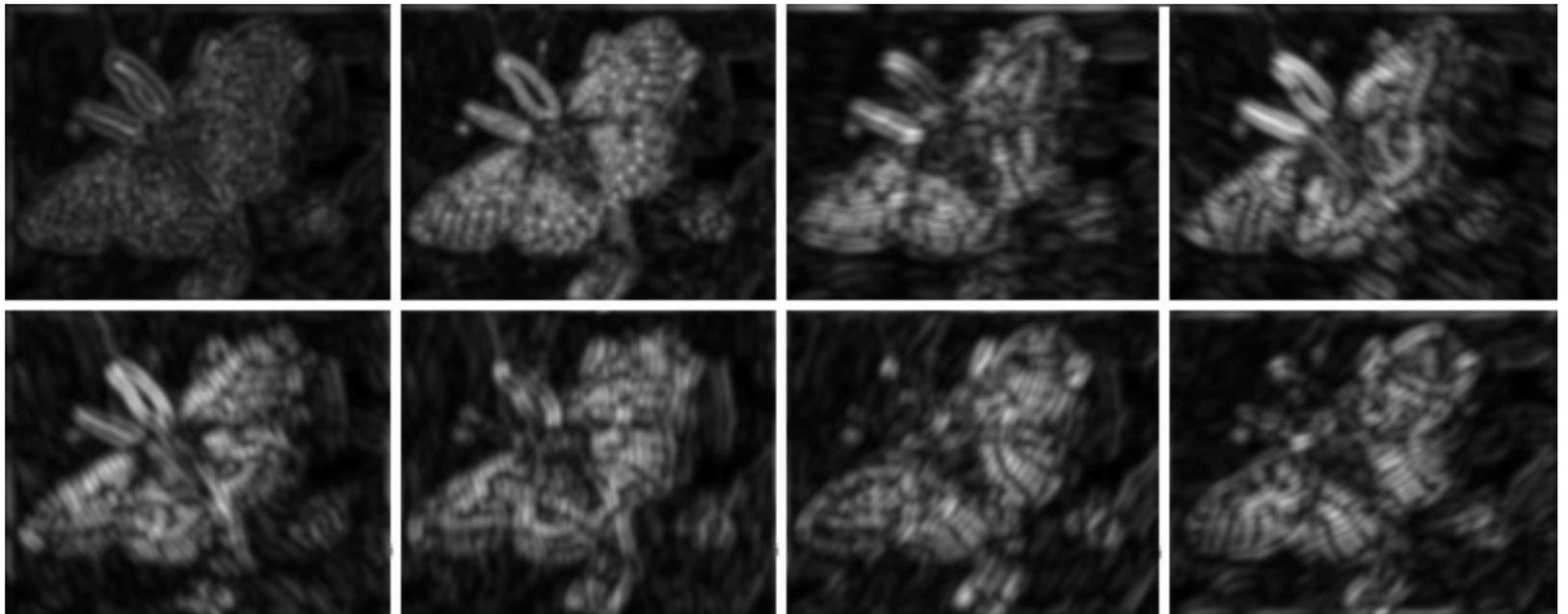
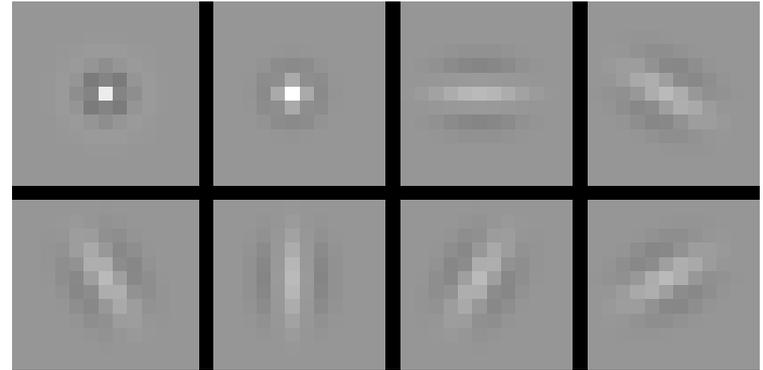
Petite échelle

# Banque de filtres sur une image

*Image  
d'entrée*

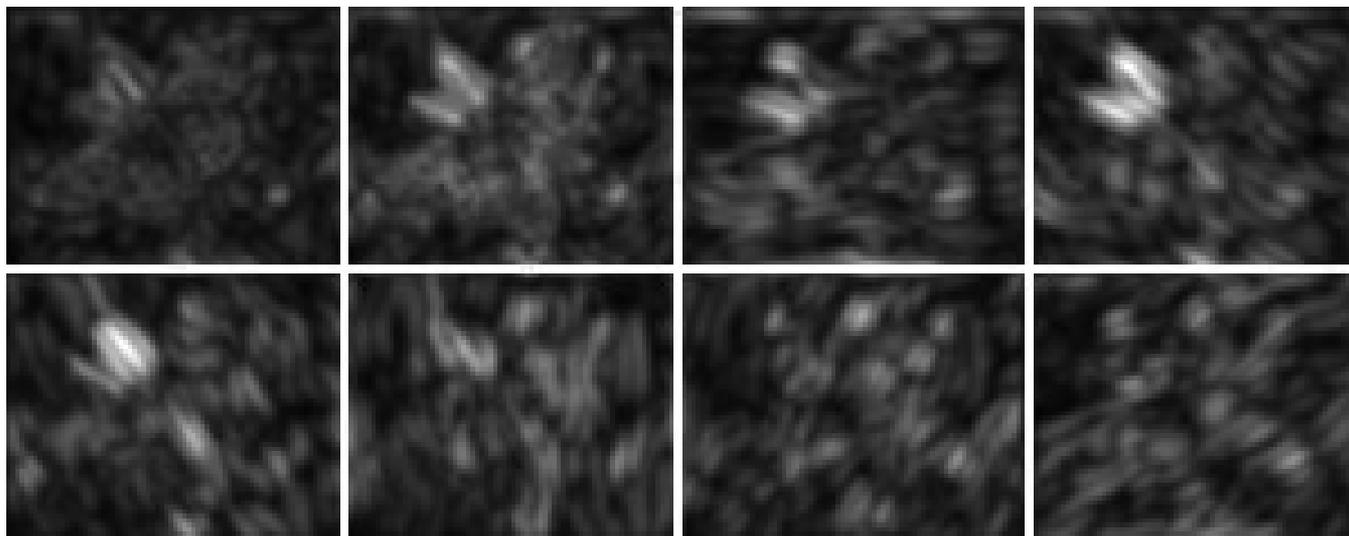
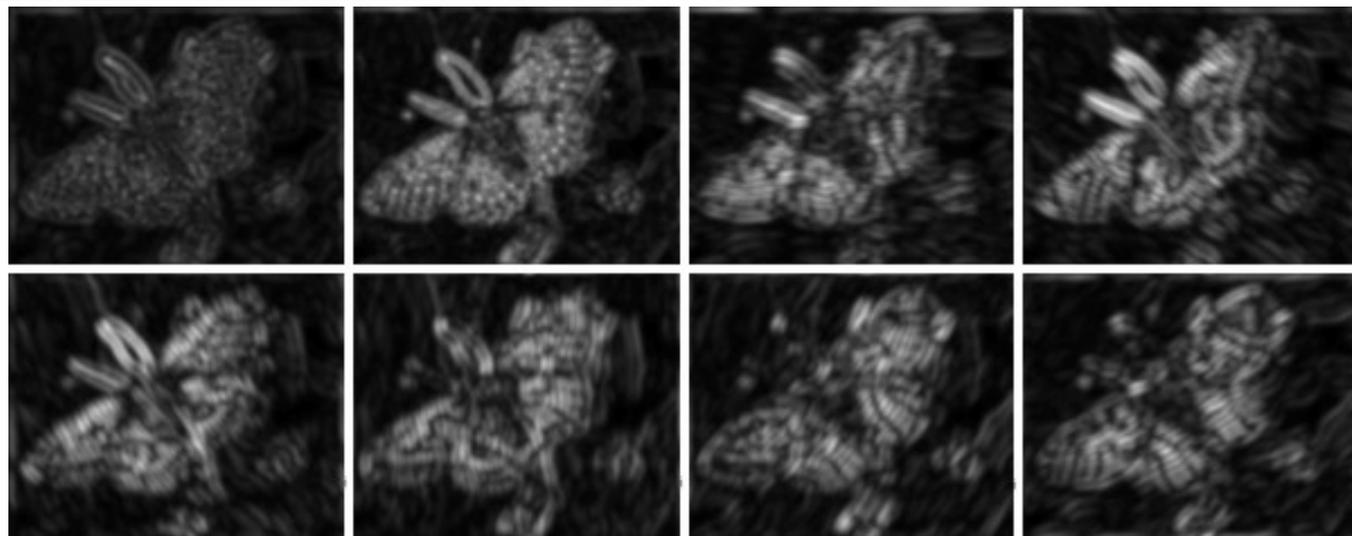


*Banque  
de filtres*



*Réponse des filtres à une échelle*

# Filtrage à différentes échelles



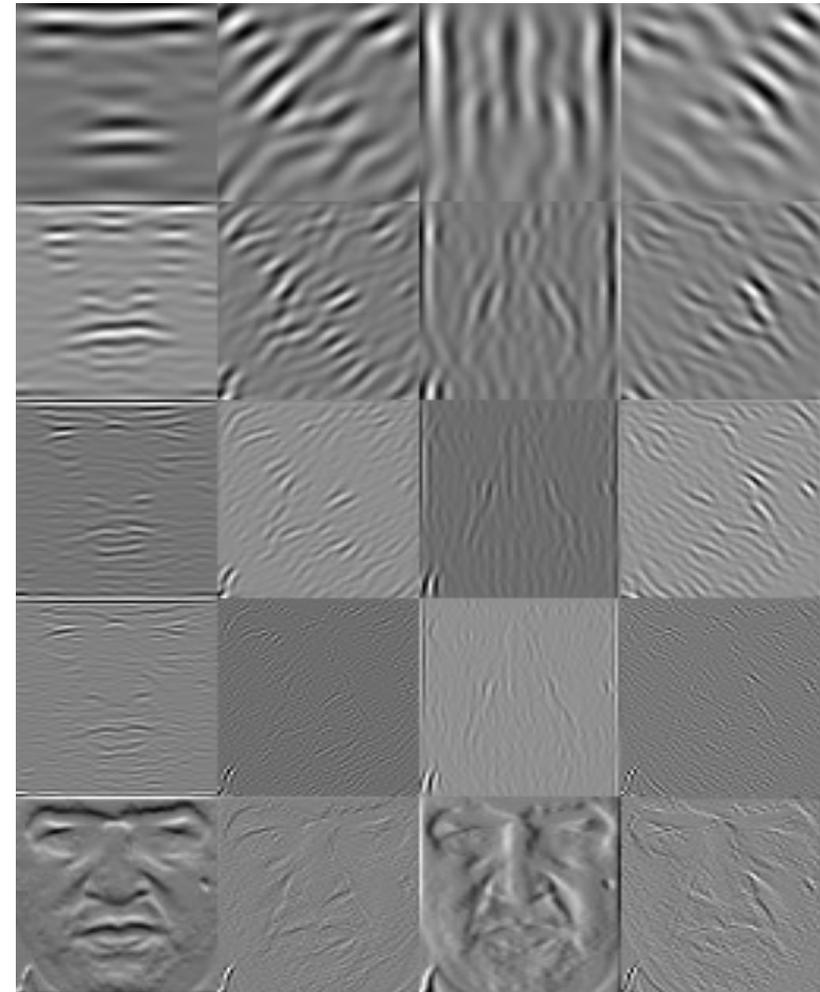
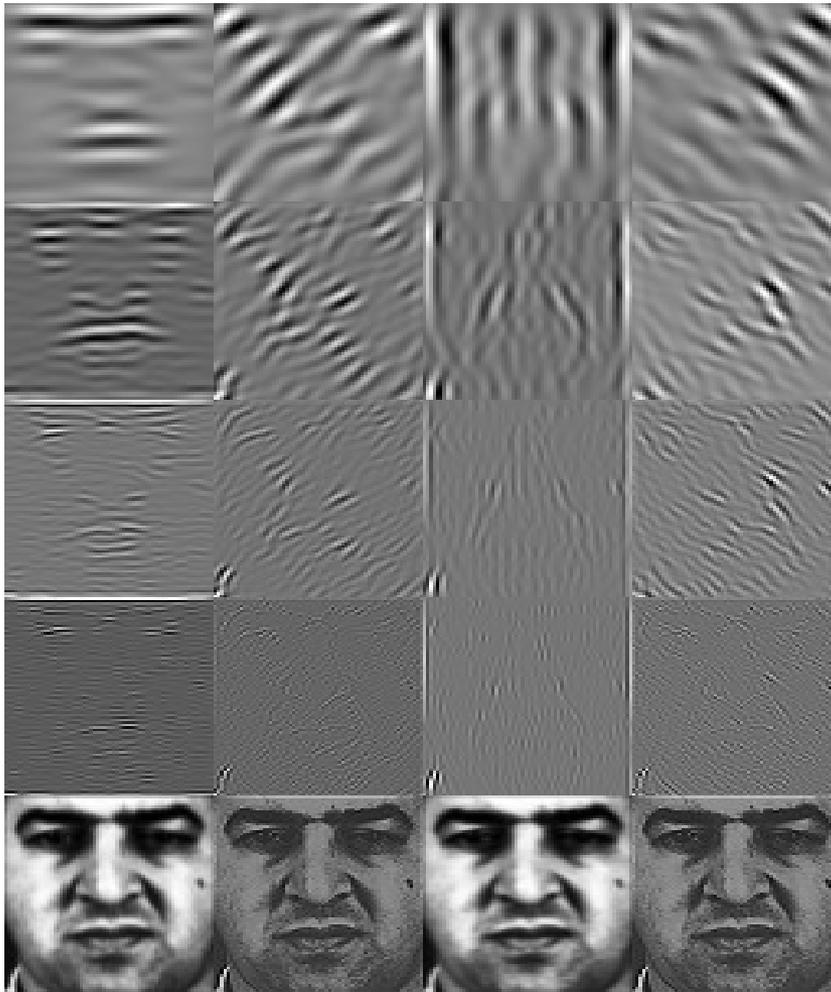
# Exemple (Torch3Vision gabor2d)

*Partie réelle*

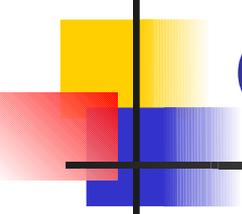
*Partie imaginaire*

(s)

*Echelle  
des  
filtres*



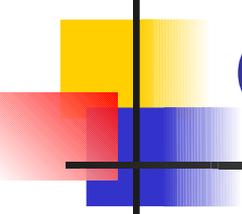
*Orientation des filtres (o)*



# Calcul de statistiques

---

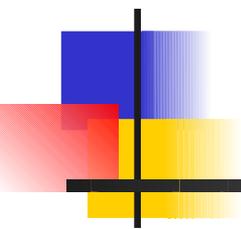
- On obtient une série de réponses aux différents filtres
  - *réponse complexe (partie réelle + partie imaginaire)*
  - *une série d'images de réponse pour chaque image de texture*
- Calcul de statistique sur la norme au carré
  - **$\text{norme}^2 = \text{réel}^2 + \text{imaginaire}^2$**
- Quelles statistiques faut-il calculer ?
  - Plus on a de statistiques, mieux c'est
  - Minimum : **moyenne de la norme<sup>2</sup>** et **écart-type de la norme<sup>2</sup>**



# Comparaison de vecteurs

---

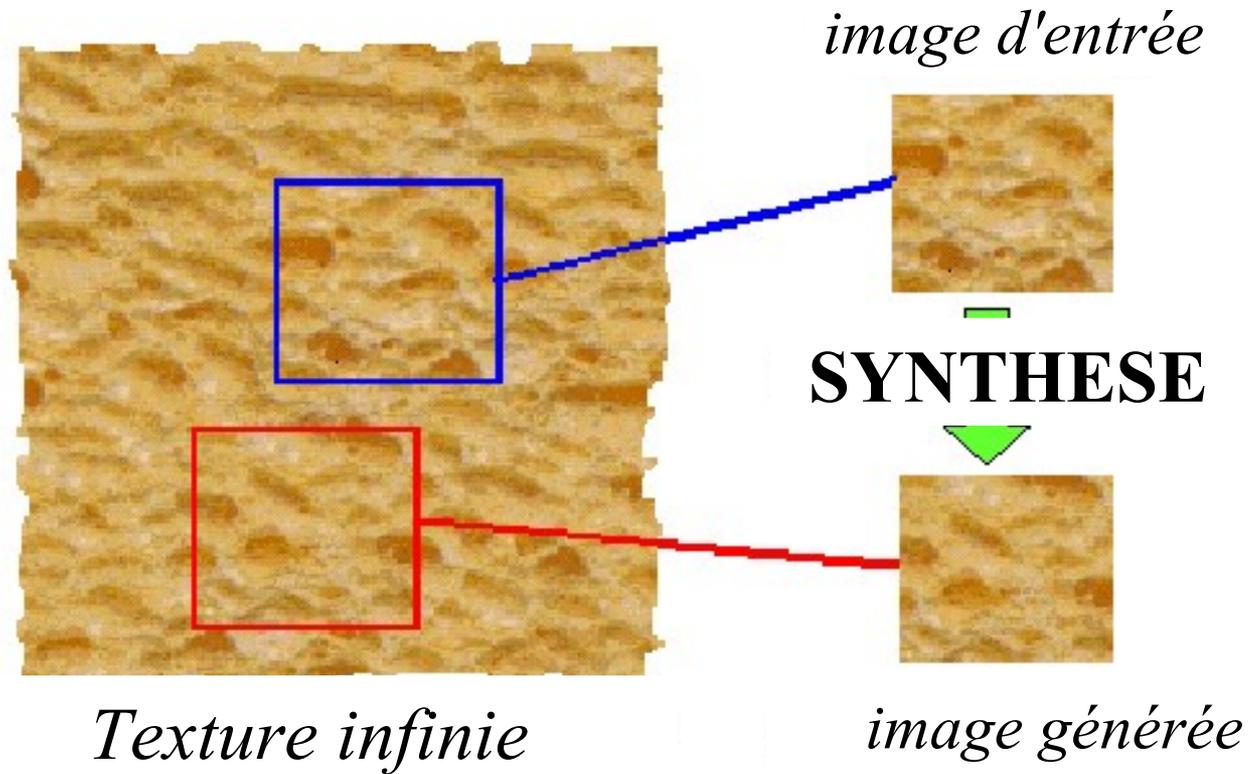
- Pour chaque image de texture, on obtient un vecteur de caractéristiques
- Comparaison de textures : calcul de distance entre les vecteurs de caractéristiques
- Distance euclidienne
  - N'est pas invariante (rotation + échelle)
- Il existe d'autres méthodes de comparaison plus efficace
  - pour obtenir l'invariance (rotation + échelle)



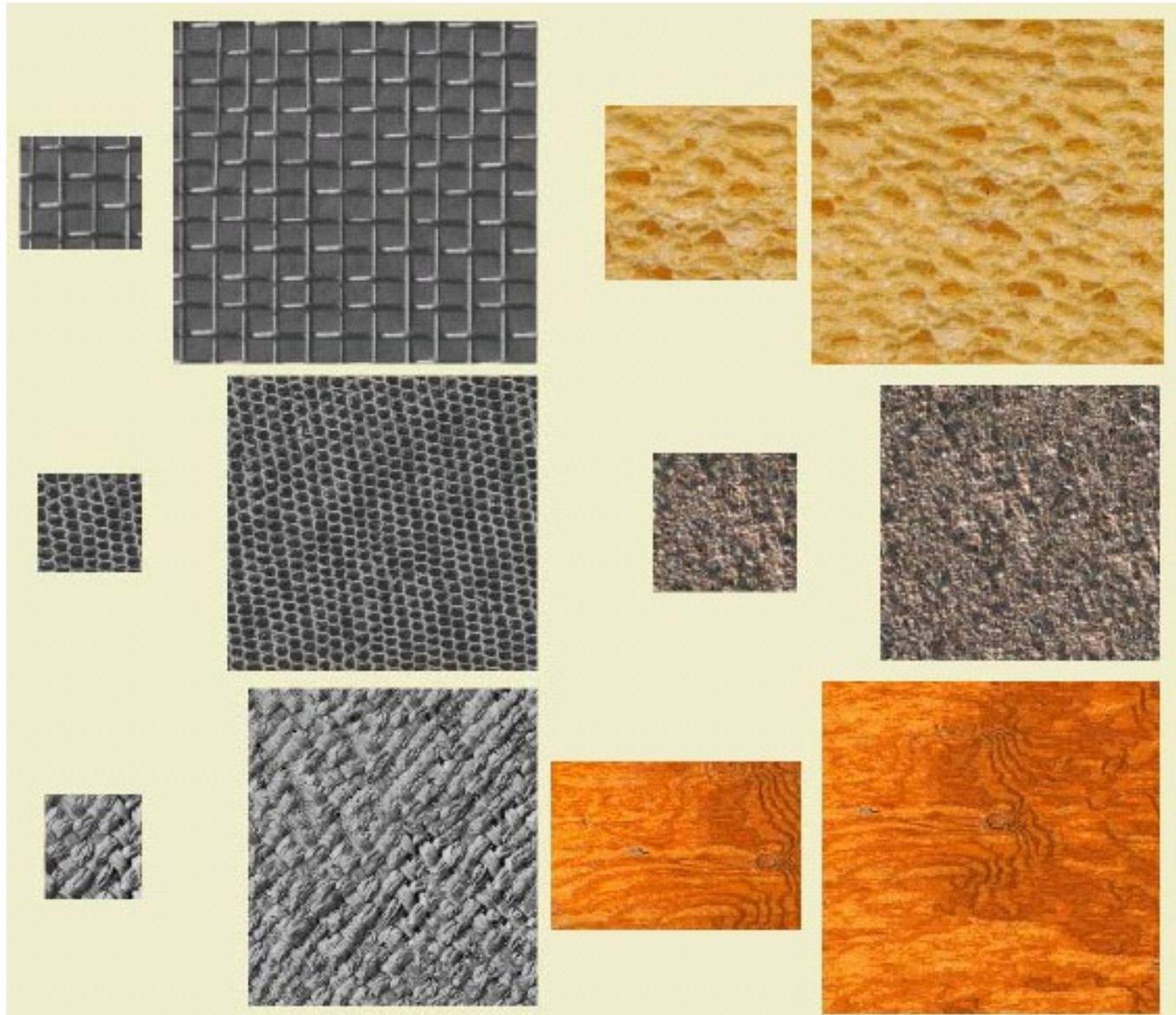
# Synthèse de texture

---

# Synthèse de texture



# Exemples (Efros et Leung)

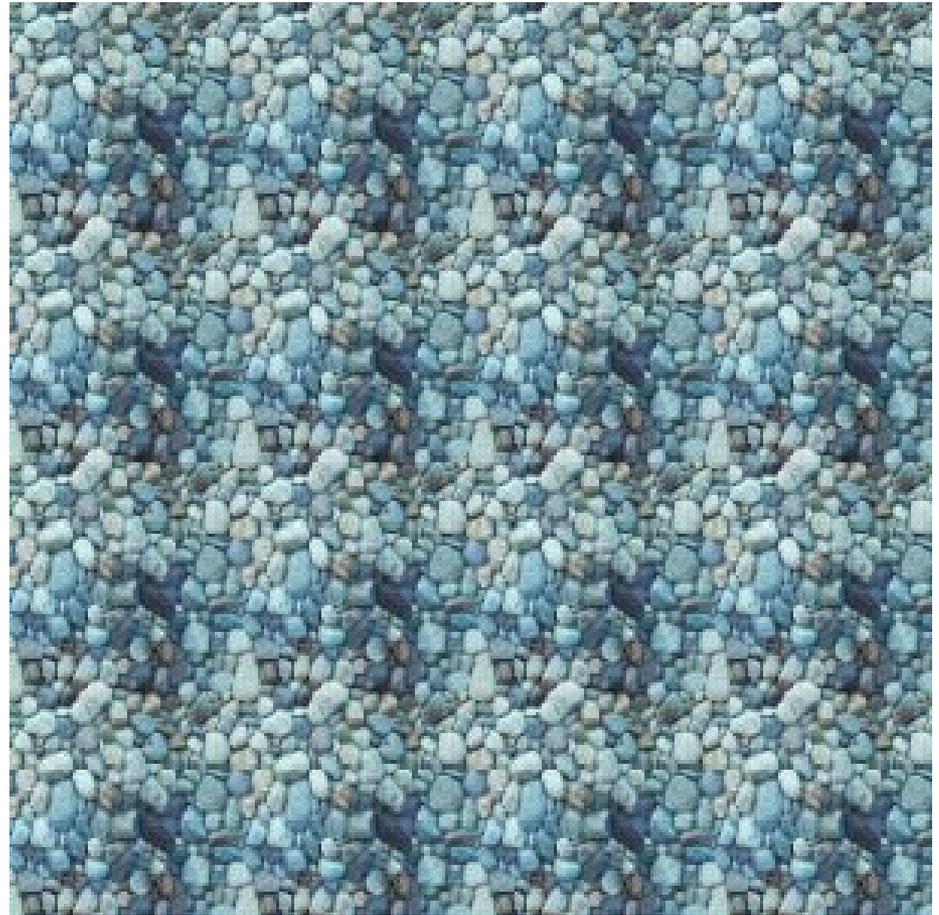


Source : David Lowe, *Texture*, CPSC 425: Computer Vision, UBC (Canada).

# Pas seulement un copier-coller



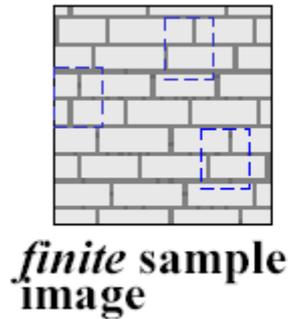
Photo



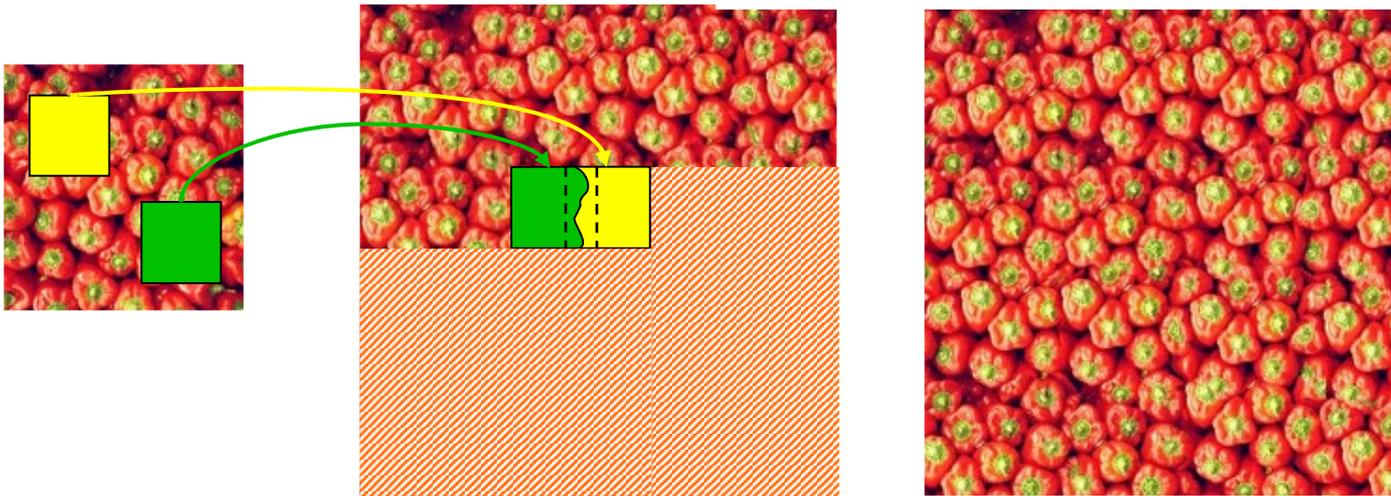
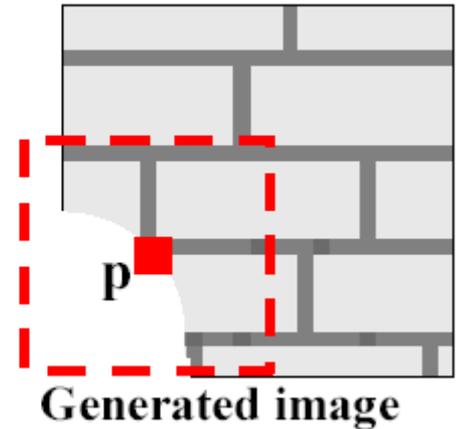
Répétition seulement

# Exemples d'algorithmes

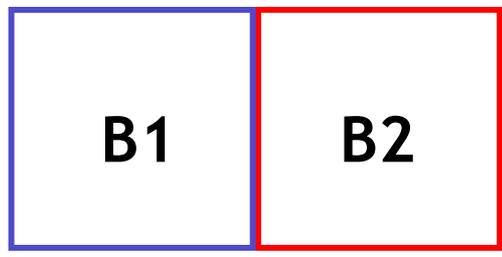
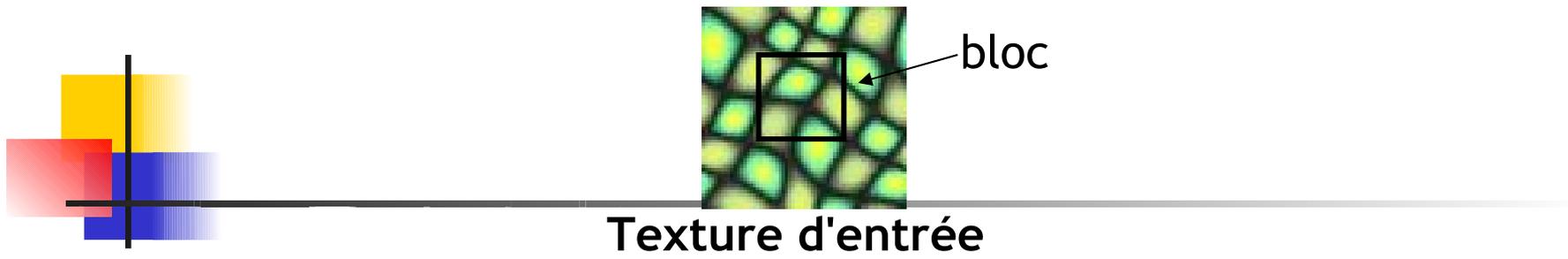
[Efros & Leung]



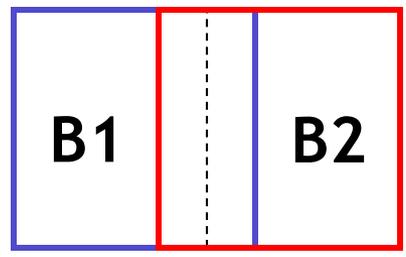
SAMPLE



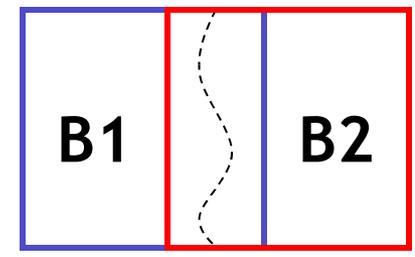
[Efros & Freeman, 2001]



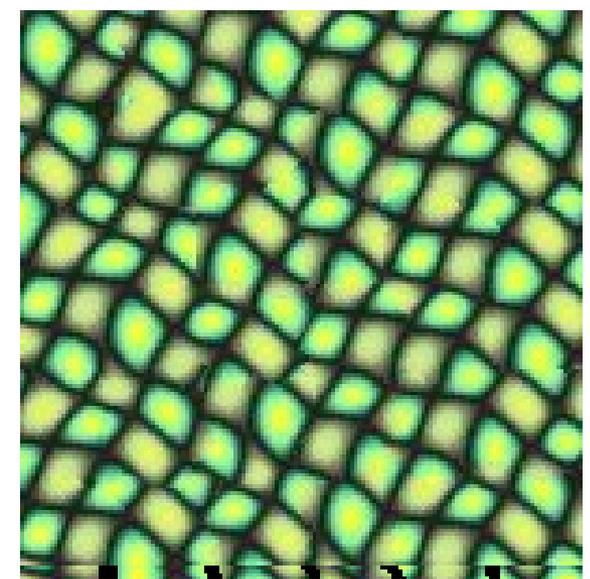
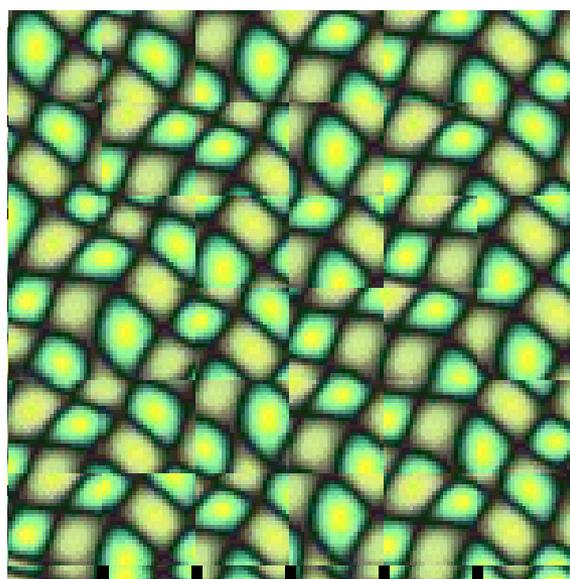
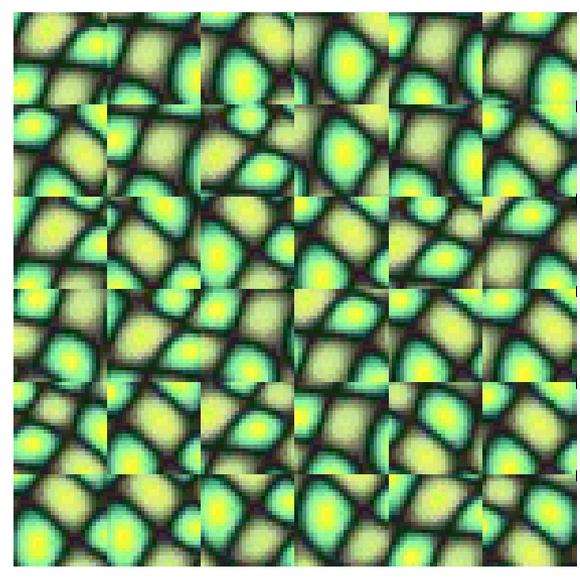
Placement aléatoire des blocs



Recouvrement des blocs voisins

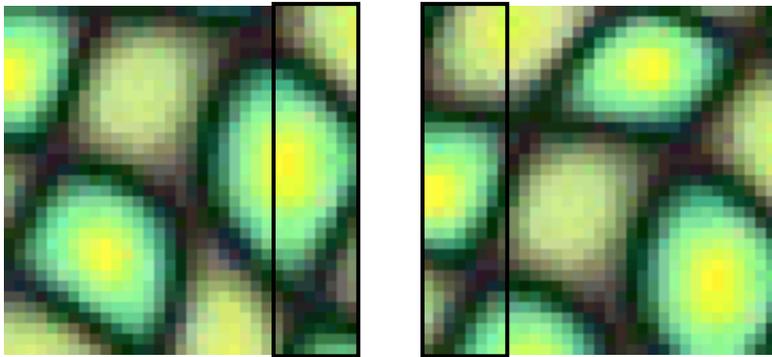


Calcul de la frontière d'erreur minimale

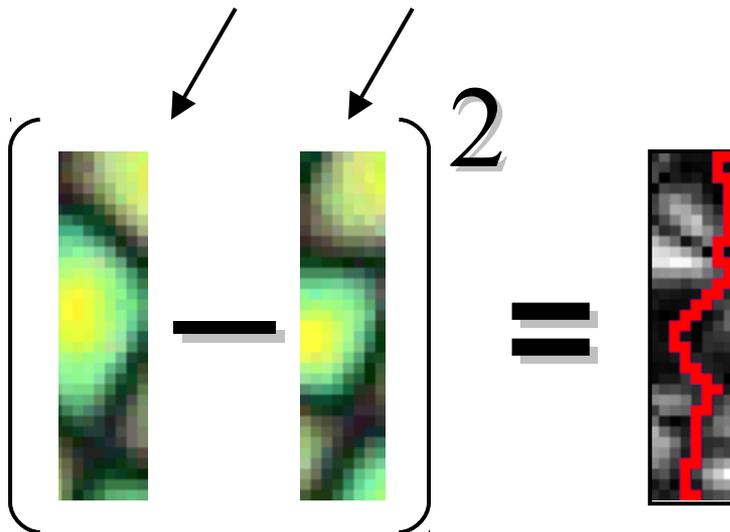
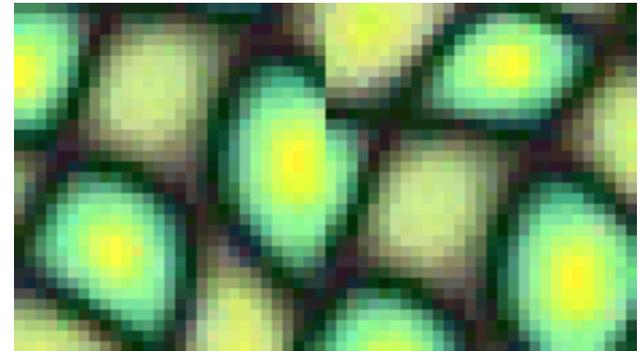


# Frontière d'erreur minimale

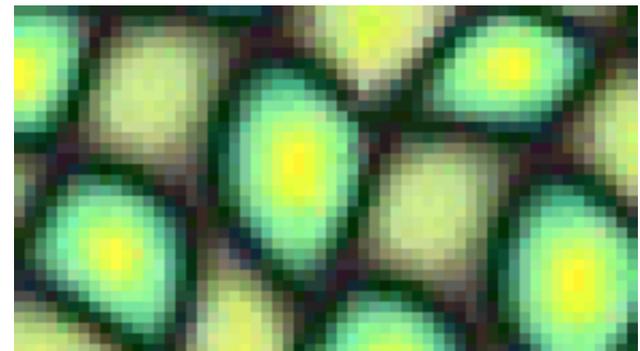
Recouvrement de blocs



Frontière verticale

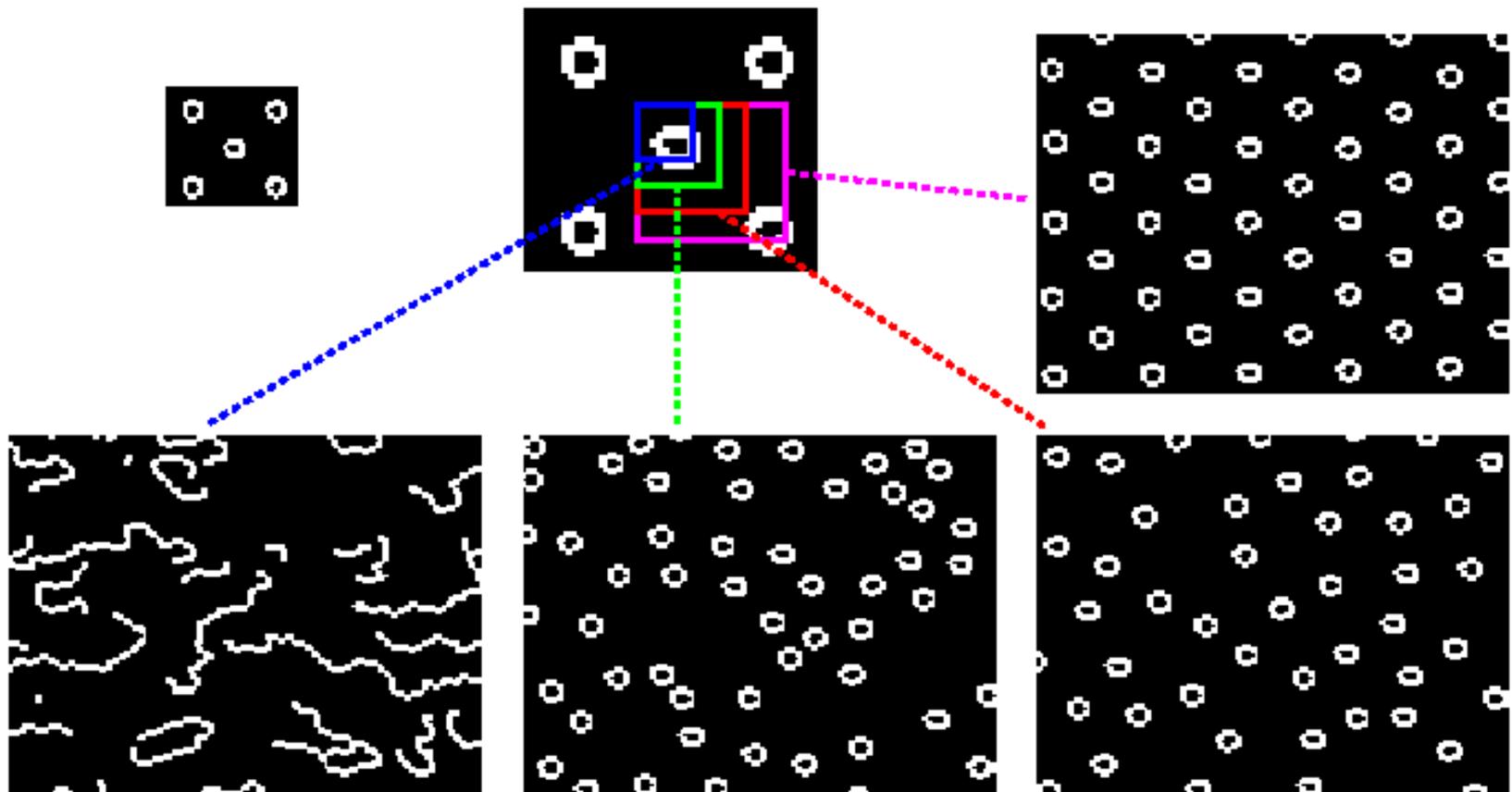


Erreur de recouvrement



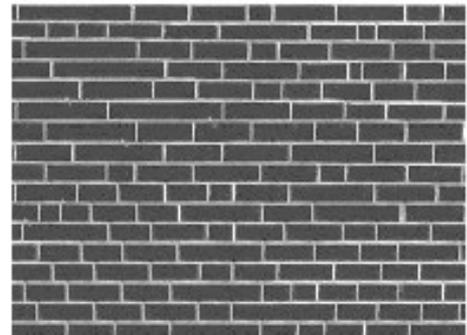
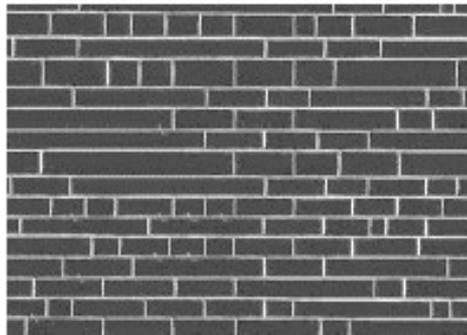
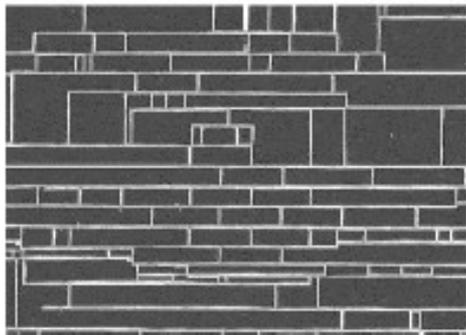
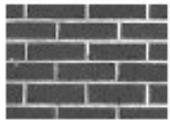
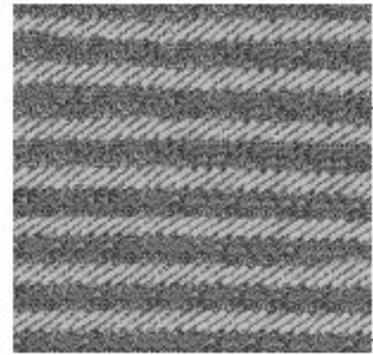
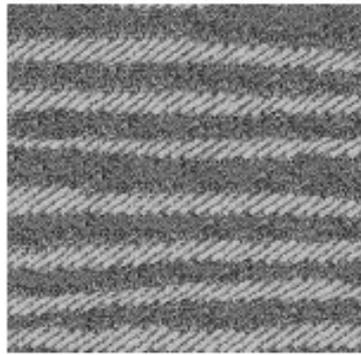
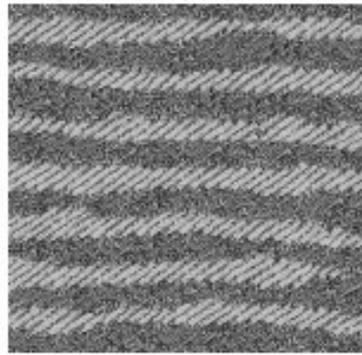
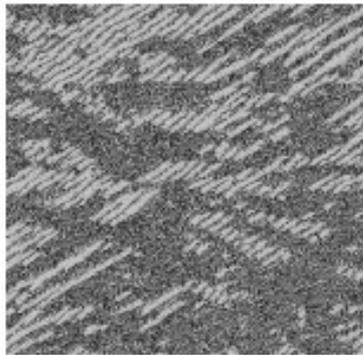
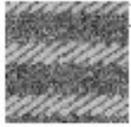
Erreur minimale

# Influence de la taille de l'échantillon



<http://www.cs.berkeley.edu/~efros/research/NPS/efros-iccv99.ppt>

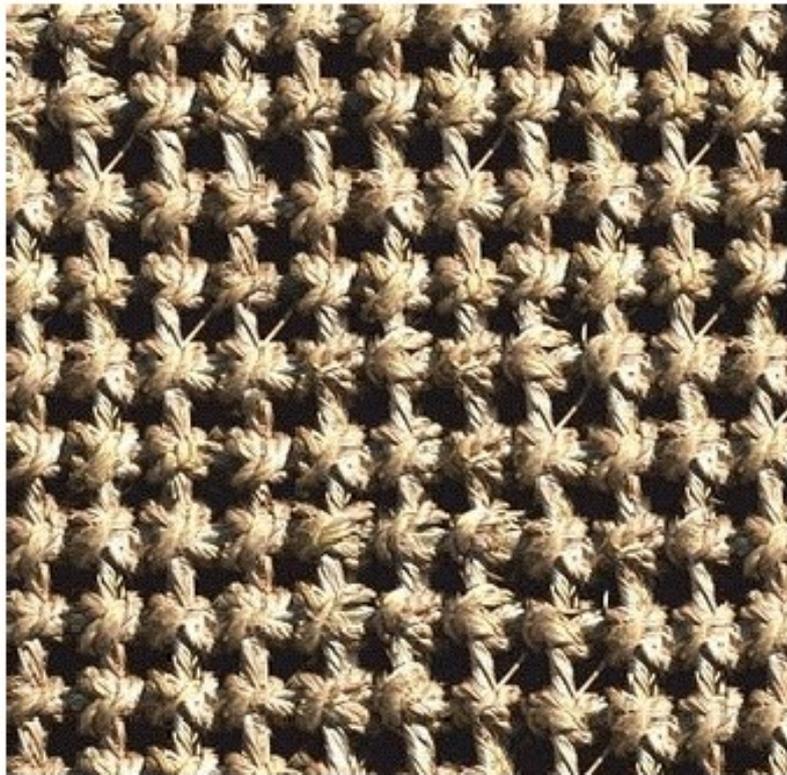
# Résultats de synthèse de texture



*Taille de l'échantillon*



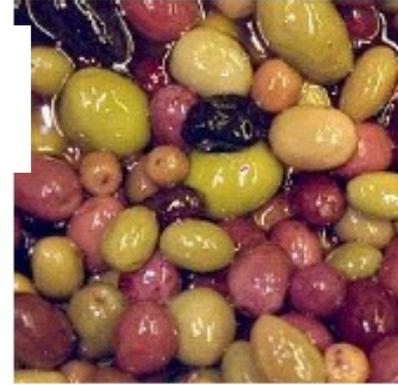
<http://www.cs.berkeley.edu/~efros/research/NPS/efros-iccv99.ppt>



Source : David Lowe, *Texture*, CPSC 425: Computer Vision, UBC (Canada).

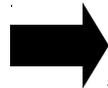


# Problèmes



# Synthèse de texte à partir d'images

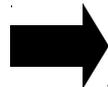
ut it becomes harder to lau  
ound itself, at "this daily  
ving rooms," as House Den  
scribed it last fall. He fail  
ut he left a ringing question  
ore years of Monica Lewin  
inda Tripp?" That now seen  
Political comedian Al Fran  
ext phase of the story will



ut it becomes harder to lau  
ound itself, at "this daily  
ving rooms," as House Den  
scribed it last fall. He fail  
ut he left a ringing question  
ore years of Monica Lewin  
inda Tripp?" That now seen  
Political comedian Al Fran  
ext phase of the story will

ut it becomes harder to lau  
ound itself, at "this daily  
ving rooms," as House Den  
scribed it last fall. He fail  
ut he left a ringing question  
ore years of Monica Lewin  
inda Tripp?" That now seen  
Political comedian Al Fran  
ext phase of the story will

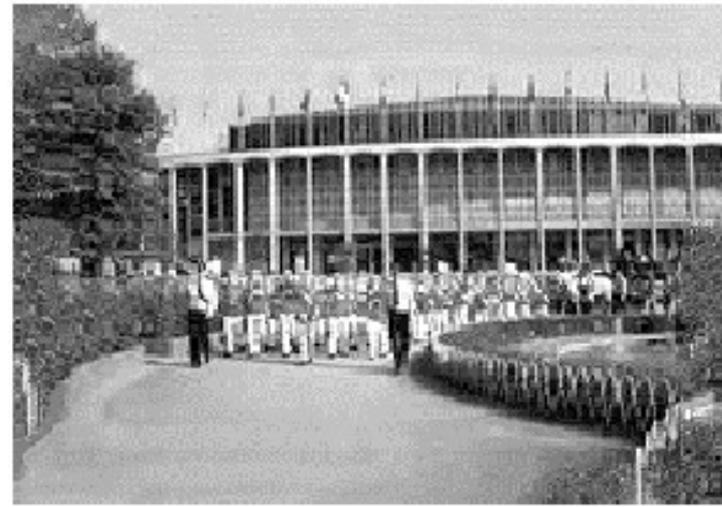
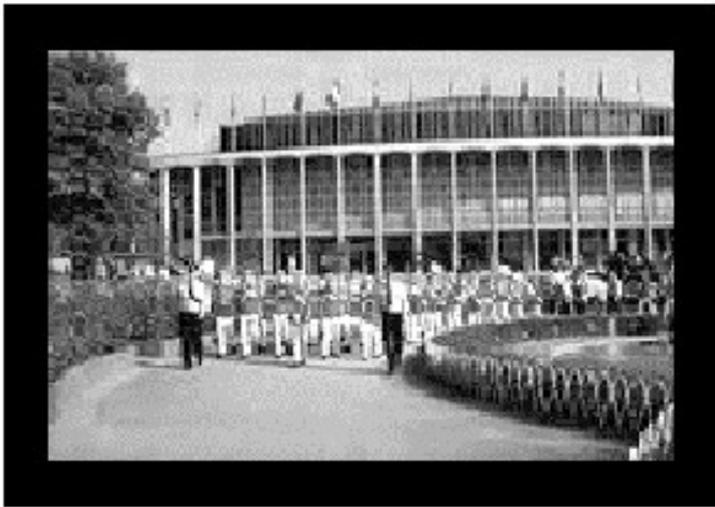
oming in the unsensatio  
r Dick Gephardt was fai  
rful riff on the looming  
nly asked, "What's your  
tions?" A heartfelt sigh  
story about the emergene  
es against Clinton. "Bo  
g people about continuin  
ardt began, patiently obs  
s, that the legal system h  
g with this latest tanger

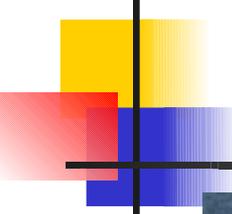


oming in the unsensatio  
r Dick Gephardt was fai  
rful riff on the looming  
nly asked, "What's your  
tions?" A heartfelt sigh  
story about the emergene  
es against Clinton. "Bo  
g people about continuin  
ardt began, patiently obs  
s, that the legal system h  
g with this latest tanger

oming in the unsensatio  
r Dick Gephardt was fai  
rful riff on the looming  
nly asked, "What's your  
tions?" A heartfelt sigh  
story about the emergene  
es against Clinton. "Bo  
g people about continuin  
ardt began, patiently obs  
s, that the legal system h  
g with this latest tanger

# Extrapolation d'images



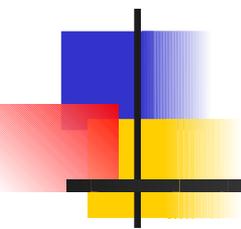


# Extrapolation d'images



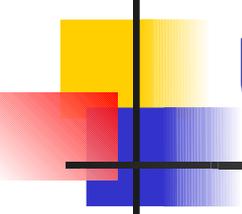
# Extrapolation d'images





# Shape from texture

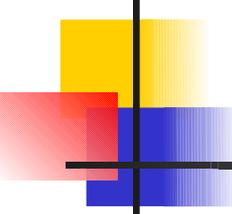
---



# Utilisation des textures

---

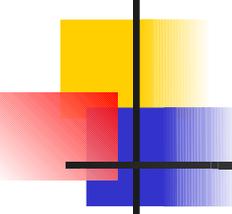
- Segmentation en régions texturées
- Caractérisation des images
  - recherche d'images par le contenu
- En théorie, il est possible de retrouver la forme 3D d'un objet à partir de sa texture (régulière)
  - *Shape from Texture*
  - En pratique, très difficile et peu d'approches existent à ce sujet
- La texture est un élément essentiel de l'image, mais souvent difficile à manipuler



# Shape from texture

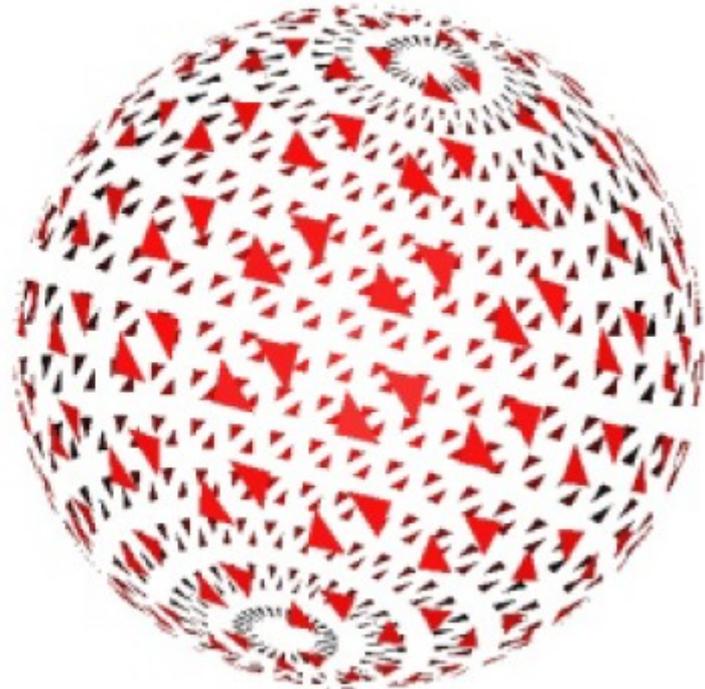
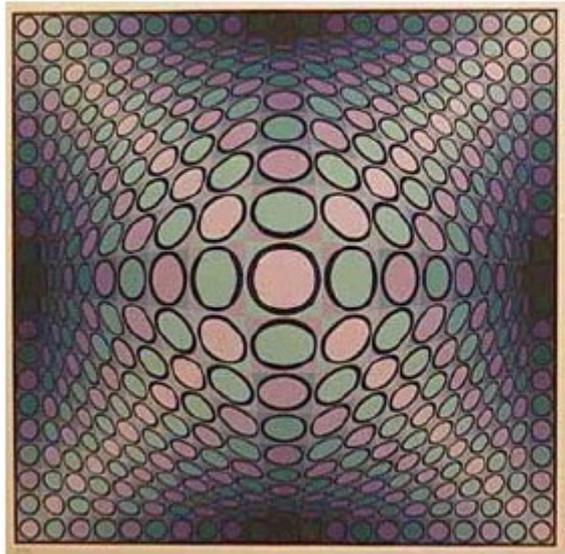
*Même texture  
mais avec une  
orientation ou une  
taille différentes*



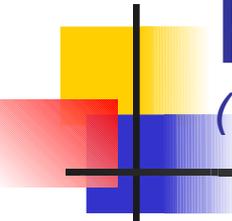


# Shape from texture

---



*Retrouvez la forme 3D d'un objet à l'aide de la texture - très difficile...*

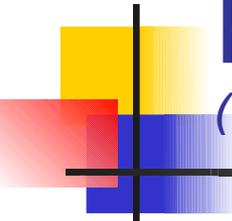


# Références

*(voir aussi la page web du cours)*

---

- Stina Svensson, Representation and Description, Computerized image analysis, Swedish University of Agricultural Sciences.
  - [www.cb.uu.se/~stina/bildSLU/lectures/F7\\_8vt2005.pdf](http://www.cb.uu.se/~stina/bildSLU/lectures/F7_8vt2005.pdf)
- Cornelia Fermüller, Lecture 24: Texture, CMSC 426 Image Processing (Computer Vision), University of Maryland at College Park.
  - <http://www.cfar.umd.edu/~fer/cmssc426/lectures/Texture2.ppt>
- Christopher Rasmussen, Texture, CIS 489/689 Computer Vision, University of Delaware (USA).
  - [http://vision.cis.udel.edu/cv/lectures/cv19\\_texture.ppt](http://vision.cis.udel.edu/cv/lectures/cv19_texture.ppt)



# Références

*(voir aussi la page web du cours)*

---

- David Lowe, Texture, CPSC 425: Computer Vision, University of British Columbia (Canada).
  - <http://www.cs.ubc.ca/~lowe/425/slides/5-Texture.ppt>
- Frank Dellaert, Texture, CS x495 Computer Vision, Georgia Institute of Technology (USA).
  - [http://www-static.cc.gatech.edu/classes/AY2005/cs4495\\_fall/09-Texture.ppt](http://www-static.cc.gatech.edu/classes/AY2005/cs4495_fall/09-Texture.ppt)
- Marc Pollefeys, Class 7 Pyramids and textures, Comp256 Computer Vision, University of North Carolina at Chapel Hill (USA).
  - <http://www.cs.unc.edu/Research/vision/comp256/vision07.ppt>