IFT3355: Infographie
Textures et aliassage

© Pierre Poulin, Derek Nowrouzezahrai

Dép. I.R.O.

Université de Montréal

Textures

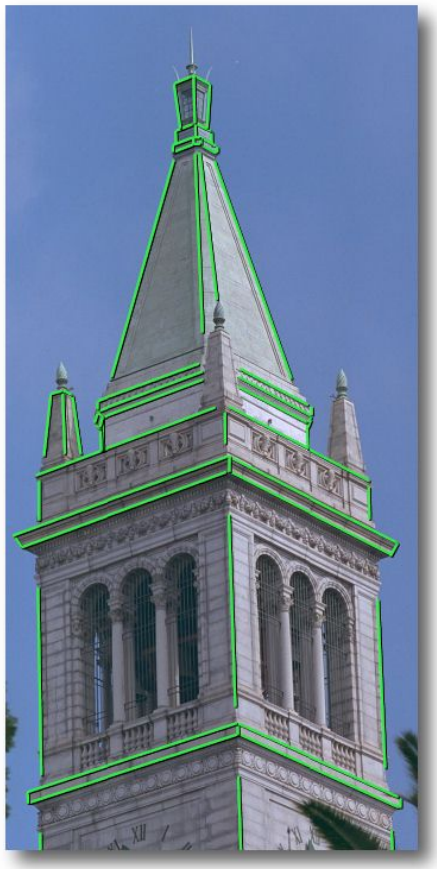
- Pour simuler la complexité des détails sur une surface, on peut augmenter le nombre de polygones (micro-polygones) définissant la surface
- Cependant, il en résulte une augmentation de la complexité de la modélisation (taille de l'objet, coût de génération, etc.) et du rendu (visibilité, *scanconversion*, illumination, etc.)

Textures

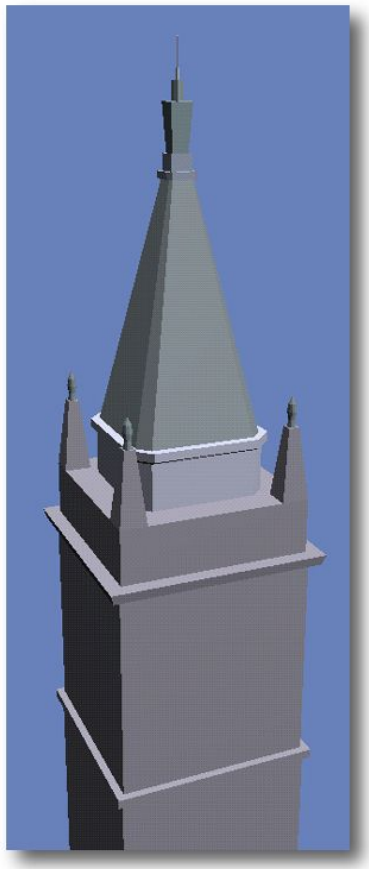
- On peut simuler l'effet visuel de ces détails en apposant une *texture* sur la surface tel du papier paint sur une surface
- Cette texture peut être apposée sur différents objets
- On laisse alors le traitement des détails au processus de *shading* dans le *pipeline* graphique
- Il devient aussi possible de mieux *filtrer* les effets de ces détails au niveau de qualité désiré dans l'image
- Hardware efficace pour les polygones texturés

Modeling and Rendering Architecture from Photographs

Debevec, Taylor, and Malik 1996

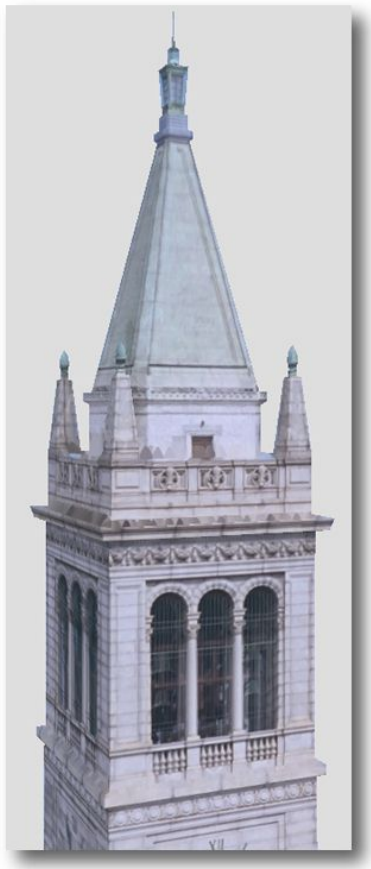


Original photograph with marked edges



Recovered model

Model edges projected onto photograph



Synthetic rendering

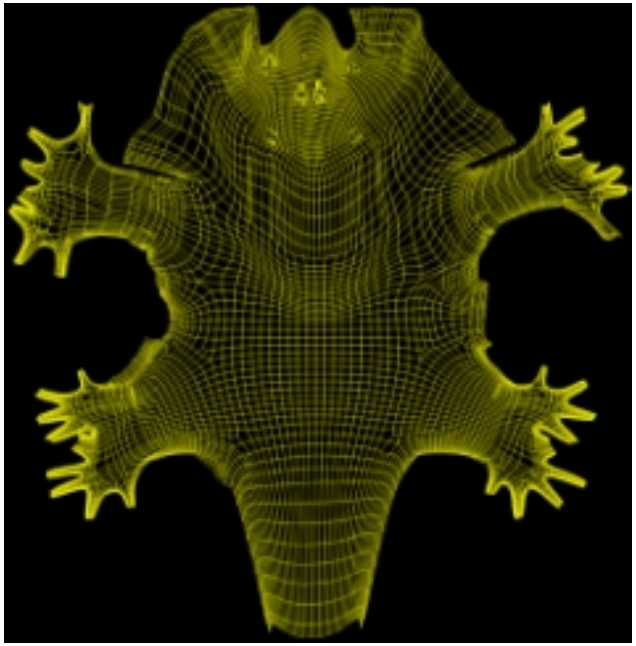
Applications de textures

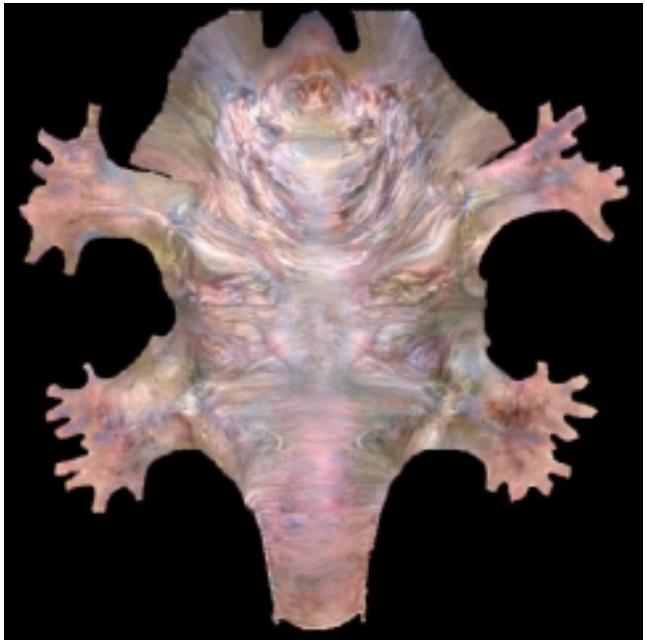
Une texture peut altérer plusieurs propriétés de surface:

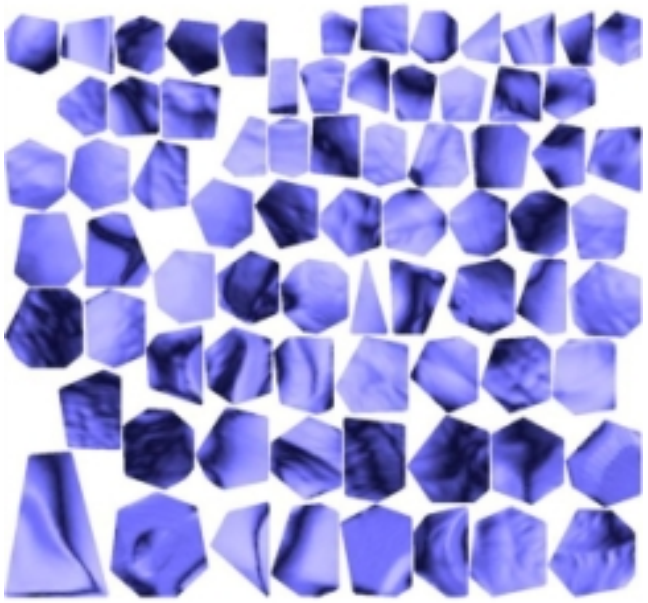
- couleur
  - image digitale, motif (quadrillé), procédure (marbre)
- normale (*bump map*)
- géométrie (*displacement map*)
- réflexion (*environment map*)
- illumination (*light map* ou *photon map*)
- transparence
- etc.

Exemples de textures

Miller







Atlas de texture

Définitions

- *Texture mapping*

  technique consistant à appliquer une texture sur une surface

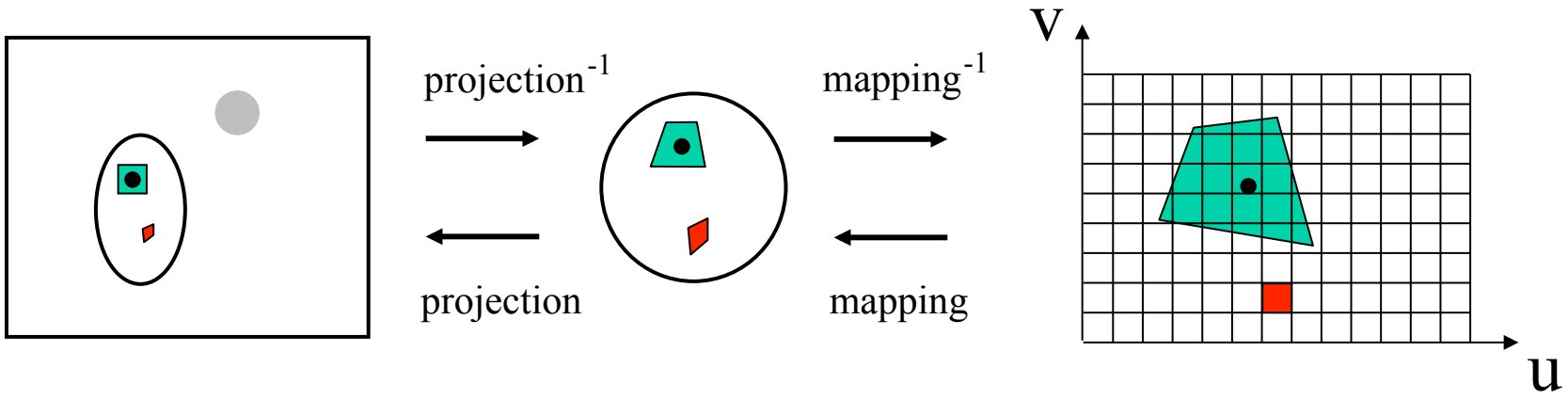
- *Texture map*

  texture apposée sur la surface

- *Texel*

  élément de base d'une texture, analogue au pixel d'une image

Texture mapping



Espace image	Espace objet	Espace texture
pixel de l'image	projection du pixel sur la surface 3D	texels couverts par le pixel

Note: discontinuité de visibilité, contours courbes

Contribution de la texture au pixel

texel ⟶ pixel

+ tous les texels sont traités dans l'ordre (cohérence)
- texel ne recouvre pas nécessairement tout un pixel. Il faut alors calculer la proportion du pixel couvert par le texel
- plus simple, mais la contribution de tout texel caché est calculée inutilement

Contribution de la texture au pixel

pixel ⟶ texel

+ seulement les pixels visibles sont traités
- il faut calculer la projection inverse du pixel en espace objet, et puis les coordonnées correspondantes en espace texture
- plus complexe, mais orienté vers le traitement du *pipeline* conventionnel qui procède pixel par pixel

Mappings

- Il faut identifier une fonction (*mapping*) qui permet d'associer un texel à un point de la surface
- On ne traite pas de point-à-point mais plutôt d'aire-à-aire, ce qui crée des problèmes d'aliassage
- Souvent on veut passer d'un rectangle à une surface 3D arbitraire
- Une fonction de mapping est souvent complexe et souffre de propriétés indésirables
  - distorsion des formes et des distances

Mappings

- Souvent un *mapping 1-to-1* est préférable
- Obtenir le *mapping* désiré n'est cependant pas une tâche facile, et il y a même des topologies qui ne permettent pas d'atteindre le but désiré

Mappings par une fonction intermédiaire

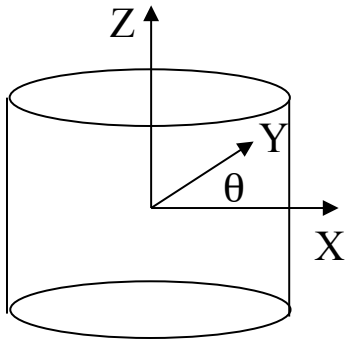
- Il existe plusieurs fonctions de défaut:
  - coordonnées cylindriques
  - coordonnées sphériques
  - coordonnées de projection
- La position de l'objet par rapport à la fonction intermédiaire définit le *mapping* et donc l'apparance de la texture sur l'objet

Mapping d'un rectangle à un cylindre

$$0 \le \theta \le 2\pi \qquad x = r\cos\theta$$

$$-1 \le h \le 1 \qquad y = r\sin\theta$$

$$z = h$$



$$0 \le \frac{\theta}{2\pi} \le 1$$

$$0 \le \frac{z+1}{2} \le 1$$

$$u = \begin{cases} \dfrac{\arccos(x/r)}{2\pi} & \text{si } y > 0 \\ 1 - \dfrac{\arccos(x/r)}{2\pi} & \text{si } y < 0 \end{cases}$$

$$v = \frac{z+1}{2}$$

ou encore

$$u = \tan^{-1}(y/x)$$

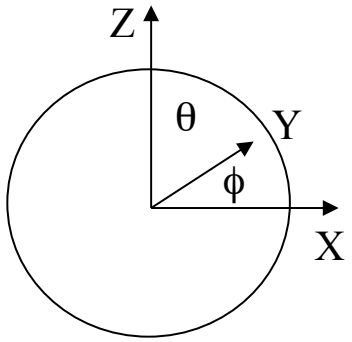
$$v = \frac{z+1}{2}$$

Mapping d'un rectangle à une sphère

$$0 \le \phi \le 2\pi \qquad x = r\sin\theta\cos\phi$$

$$0 \le \theta \le \pi \qquad y = r\sin\theta\sin\phi$$

$$z = r\cos\theta$$



$$0 \le \frac{\phi}{2\pi} \le 1$$

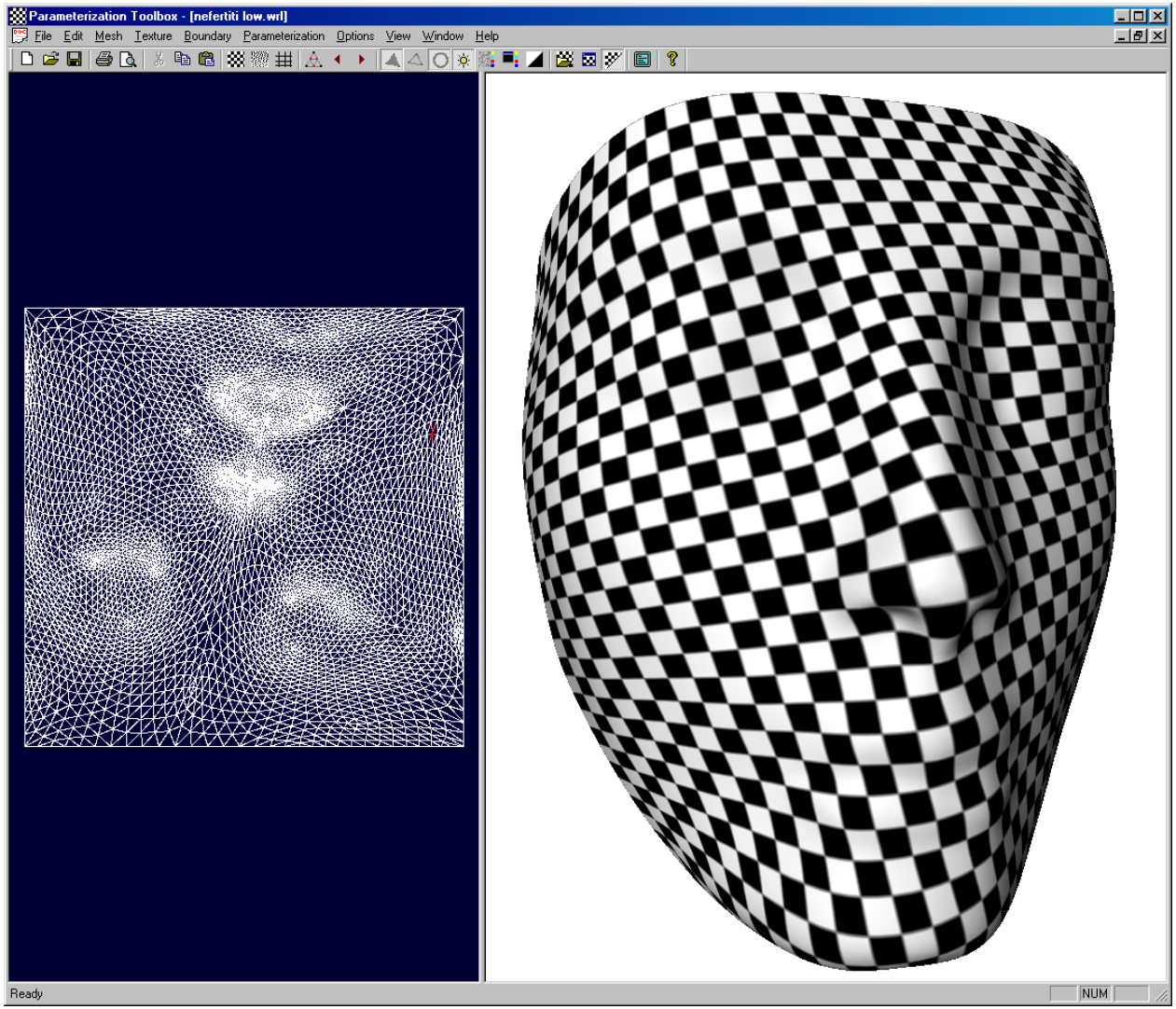
$$0 \le \frac{\theta}{\pi} \le 1$$

$$u = \begin{cases} \dfrac{\arccos(x/\sqrt{x^2+y^2})}{2\pi} & \text{si } y > 0 \\ 1 - \dfrac{\arccos(x/\sqrt{x^2+y^2})}{2\pi} & \text{si } y < 0 \end{cases}$$

$$v = \frac{\arccos(z/r)}{\pi}$$

Cyberware

Reparamétrisation pour préserver l'aire



Alliez

Tuiles (*tiling*)

texture de $n_x \times n_y$ texels

répétée s fois

texture d'indice (u,v)

Tiling

$$i = \lfloor (usn_x) \% n_x \rfloor$$

$$j = \lfloor (vsn_y) \% n_y \rfloor$$

Bilinéaire

$$\begin{aligned} c(u,v) = {} & (1-u')(1-v')c_{ij} + \\ & u'(1-v')c_{(i+1)j} + \\ & (1-u')v'c_{i(j+1)} + \\ & u'v'c_{(i+1)(j+1)} \end{aligned}$$

$$u' = usn_x - \lfloor usn_x \rfloor$$

$$v' = vsn_y - \lfloor vsn_y \rfloor$$

Hermite

$$\begin{aligned} c(u,v) = {} & (1-u'')(1-v'')c_{ij} + \\ & u''(1-v'')c_{(i+1)j} + \\ & (1-u'')v''c_{i(j+1)} + \\ & u''v''c_{(i+1)(j+1)} \end{aligned}$$

$$u'' = 3(u')^2 - 2(u')^3$$

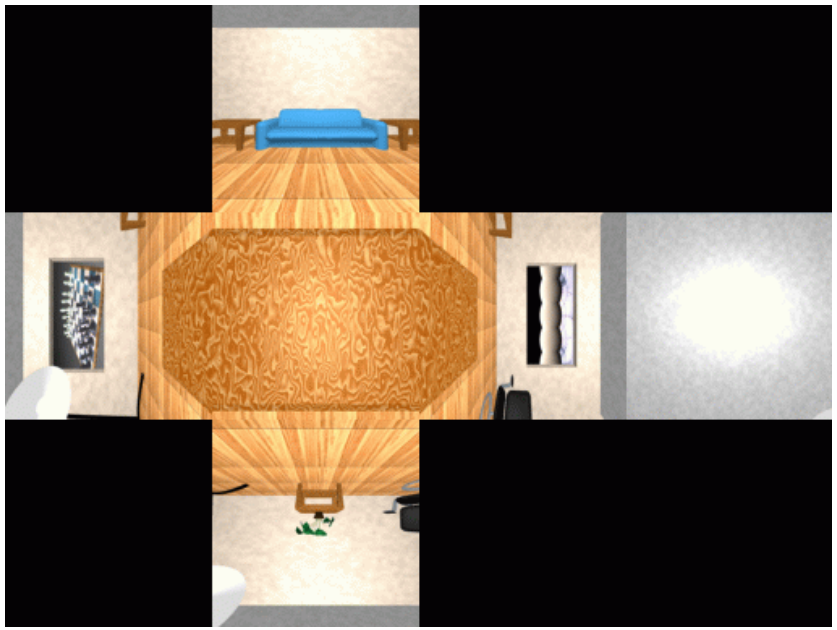
$$v'' = 3(v')^2 - 2(v')^3$$

Texture d'environnement (ou de réflexion)

- Une surface miroir réfléchit son environnement, mais le lancer de rayons est trop coûteux pour espérer du temps réel
- On crée une image de la scène (sans l'objet réfléchissant) sur un objet intermédiaire
  - cube (six projections), sphère, cylindre, etc.
- La direction réfléchie sur la surface est utilisée comme index dans les images de la scène

Environnement sphérique

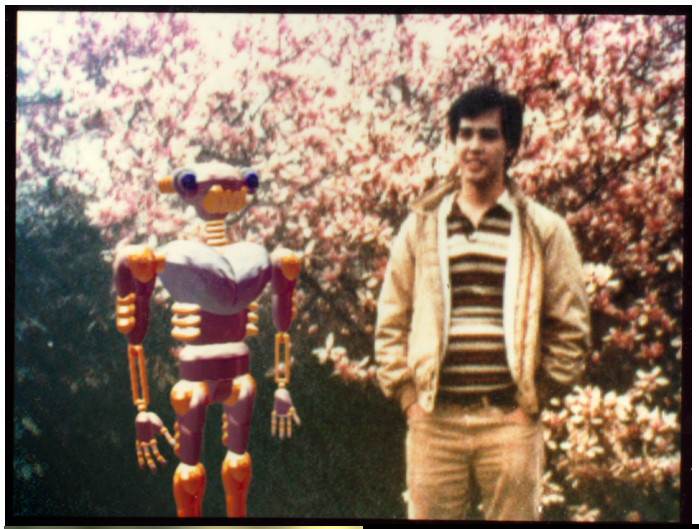
Environnement cubique

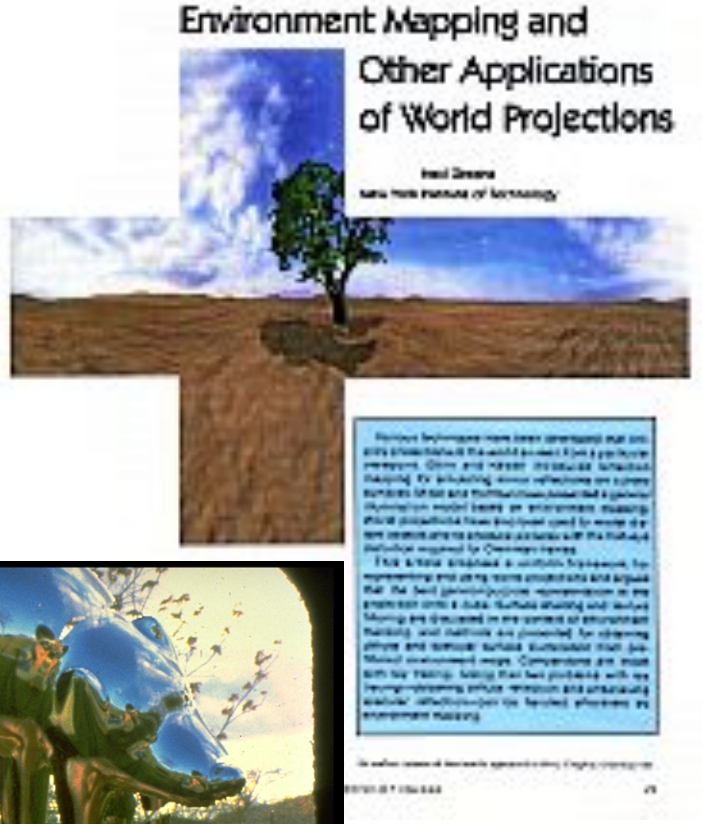


Texture d'environnement

- Aucune réflexion multiple ou réflexion de l'objet sur lui-même
- Puisque l'image n'est pas formée du même point de vue, les objets près de l'objet miroir seront incorrectements déformés dans l'image finale

Environment map



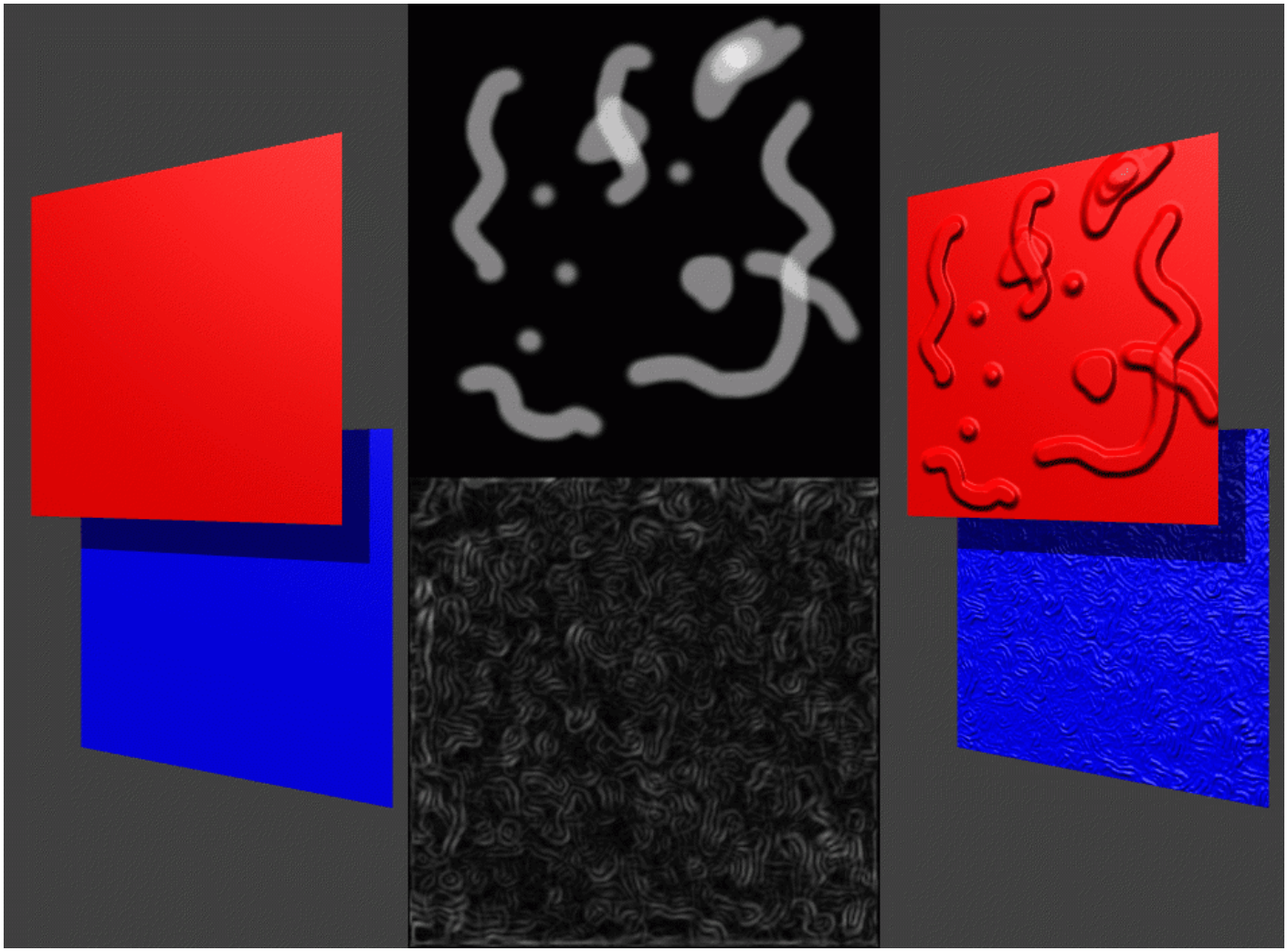


Chou

Miller

Animation Williams

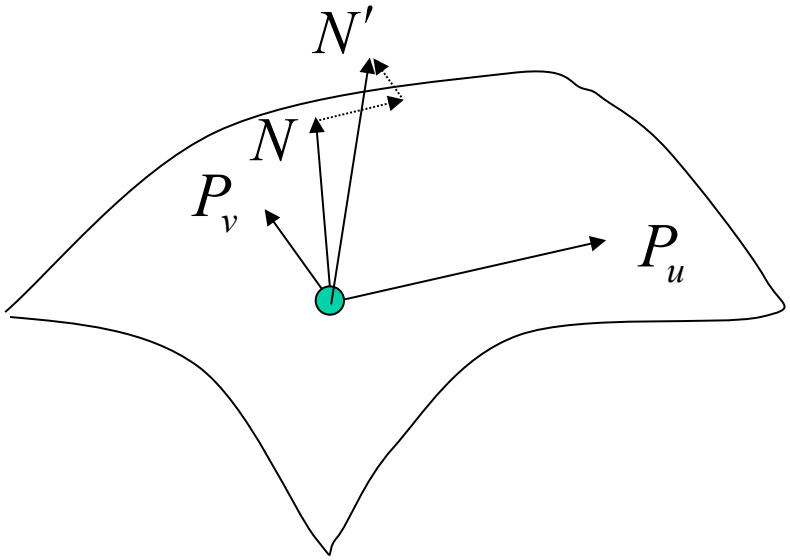
Bump map



Legakis

Bump map

- La normale à la surface influence grandement l'apparence de la surface lors du calcul d'illumination



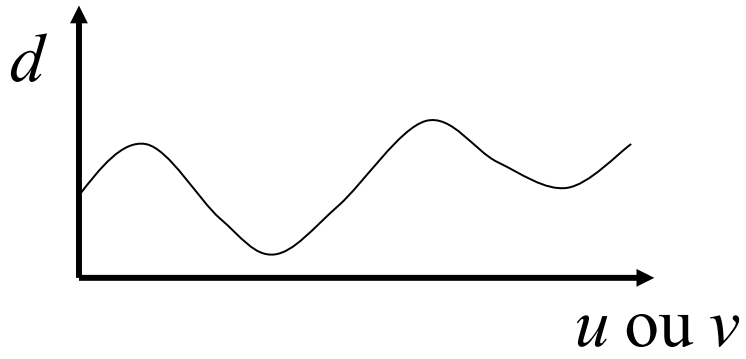
$$N = \frac{P_u \times P_v}{\left| P_u \times P_v \right|}$$

$$P_u = \begin{bmatrix} \frac{\partial x}{\partial u} \\ \frac{\partial y}{\partial u} \\ \frac{\partial z}{\partial u} \end{bmatrix}$$

$$P_v = \begin{bmatrix} \frac{\partial x}{\partial v} \\ \frac{\partial y}{\partial v} \\ \frac{\partial z}{\partial v} \end{bmatrix}$$

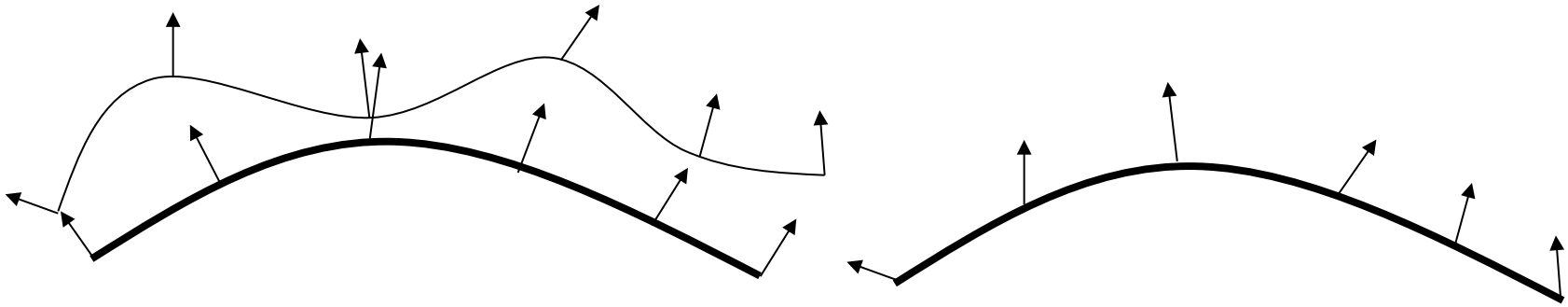
Bump map

- Soit une fonction *d(u,v)* qui perturbe légèrement un point à la surface le long de sa normale



$$P' = P + d(u,v)N$$

$$N' = \frac{P'_u \times P'_v}{\left|P'_u \times P'_v\right|}$$



Bump map

$$P'_u = P_u + \frac{\partial d}{\partial u} N + d(u,v) N_u$$

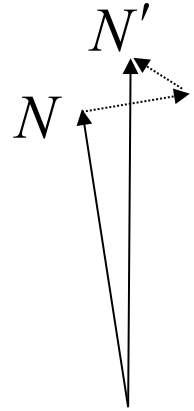
$$P'_v = P_v + \frac{\partial d}{\partial v} N + d(u,v) N_v$$

$$N \times N = 0$$

$$\left| d(u,v) \right| << 1$$

$$N' = \left(P_u \times P_v\right) + \left(P_u \times \frac{\partial d}{\partial v} N\right) + \left(P_u \times d(u,v) N_v\right) +$$
$$\left(\frac{\partial d}{\partial u} N \times P_v\right) + \left(\frac{\partial d}{\partial u} N \times \frac{\partial d}{\partial v} N\right) + \left(\frac{\partial d}{\partial u} N \times d(u,v) N_v\right) +$$
$$\left(d(u,v) N_u \times P_v\right) + \left(d(u,v) N_u \times \frac{\partial d}{\partial v} N\right) + \left(d(u,v) N_u \times d(u,v) N_v\right)$$

$$N' \approx N + \frac{\partial d}{\partial u} N \times P_v + \frac{\partial d}{\partial v} N \times P_u$$

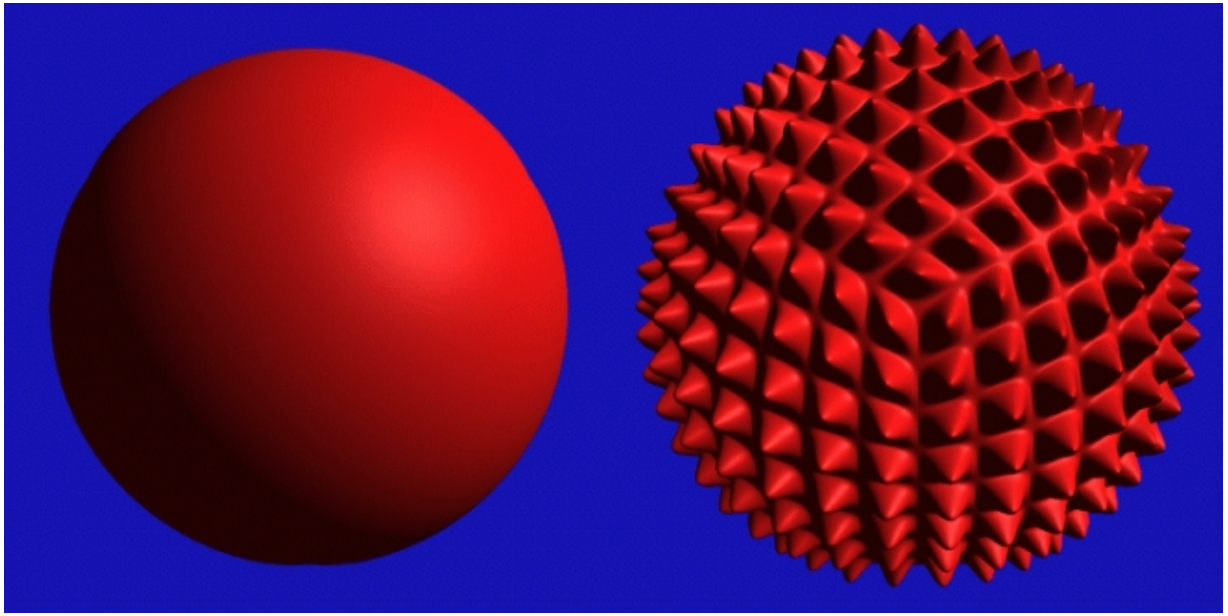


Précalcul des deux dérivées partielles

Displacement map

- Déplace la surface d'une distance le long de la normale à la surface
- Polygonise finement la surface

\+ Silhouettes et ombres projetées



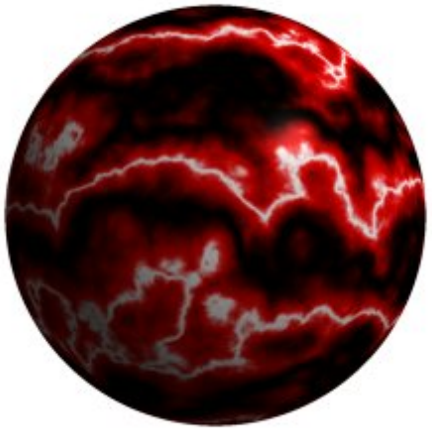
Problèmes avec le mapping 3D à 2D

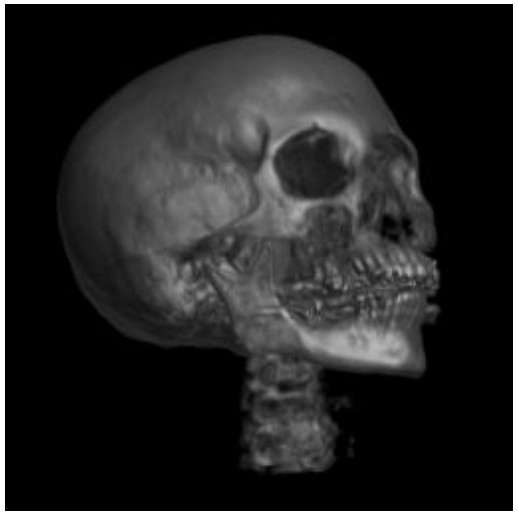
- Définition de la fonction de *mapping*
- Conservation des distances entre les éléments de la texture
- Effet de bordure de la texture lorsqu'elle doit être répétée
- Donne une impression de papier paint au lieu de matériel 3D

Textures 3D (*solid textures*)

- Par défaut, chaque point 3D peut correspondre à un point dans la texture
- Textures
  - table 3D de valeurs discrètes
  - fonction 3D (procédurale) définit une valeur *UVW* de texture dans l'espace *XYZ*
- Surface apparaît comme sculptée dans un matériel
  - marbre, bois, etc.
- Un problème important de ces textures 3D réside dans son filtrage efficace

Textures 3D

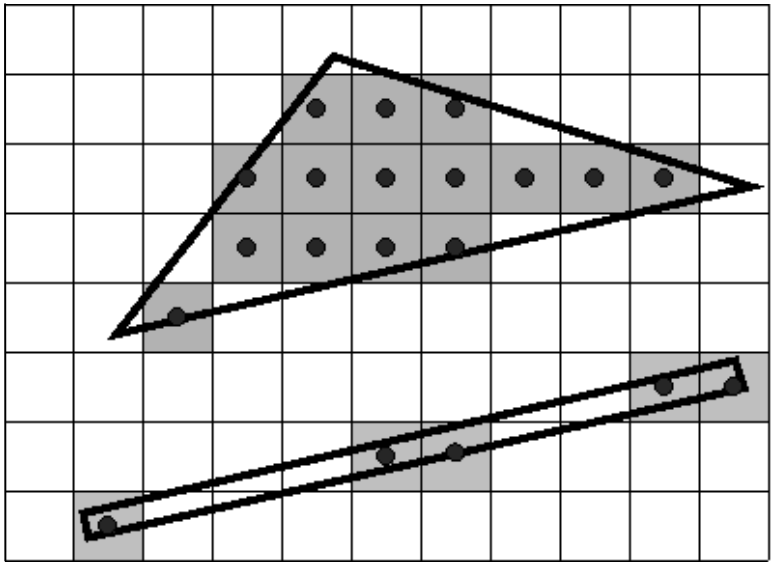


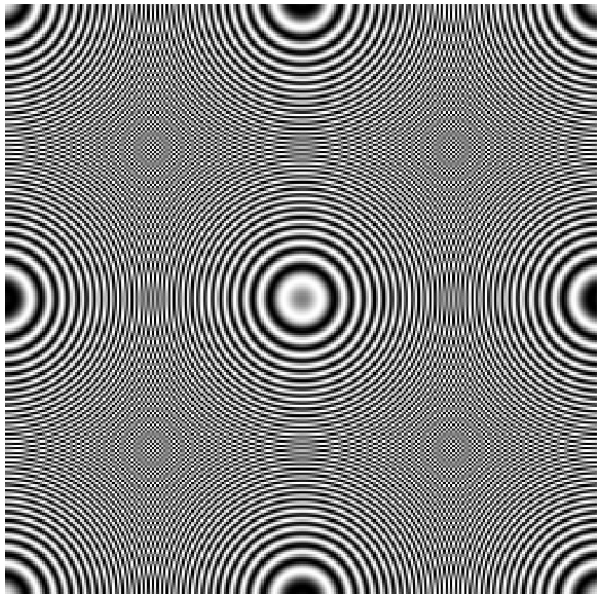


Aliassage

Les effets d'aliassage se présentent sous forme de:

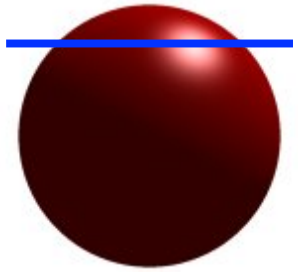
- marches d'escalier (crénelage)
- Moirés (fantômes)



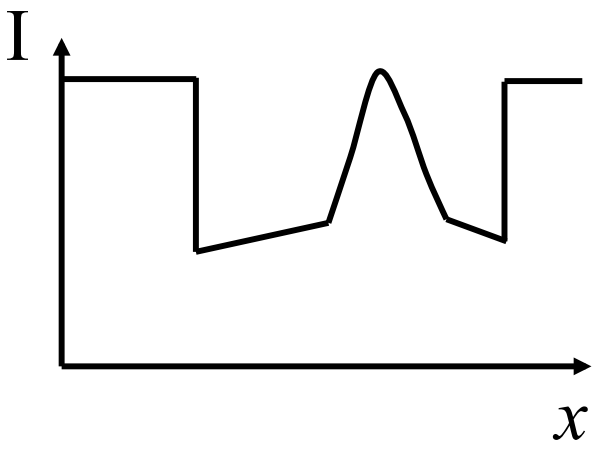


Signal dans une fenêtre

- Une scène projetée correspond à une variation d'intensité (couleur) dans l'espace (domaine spatial)



ligne d'intensité
à travers l'image

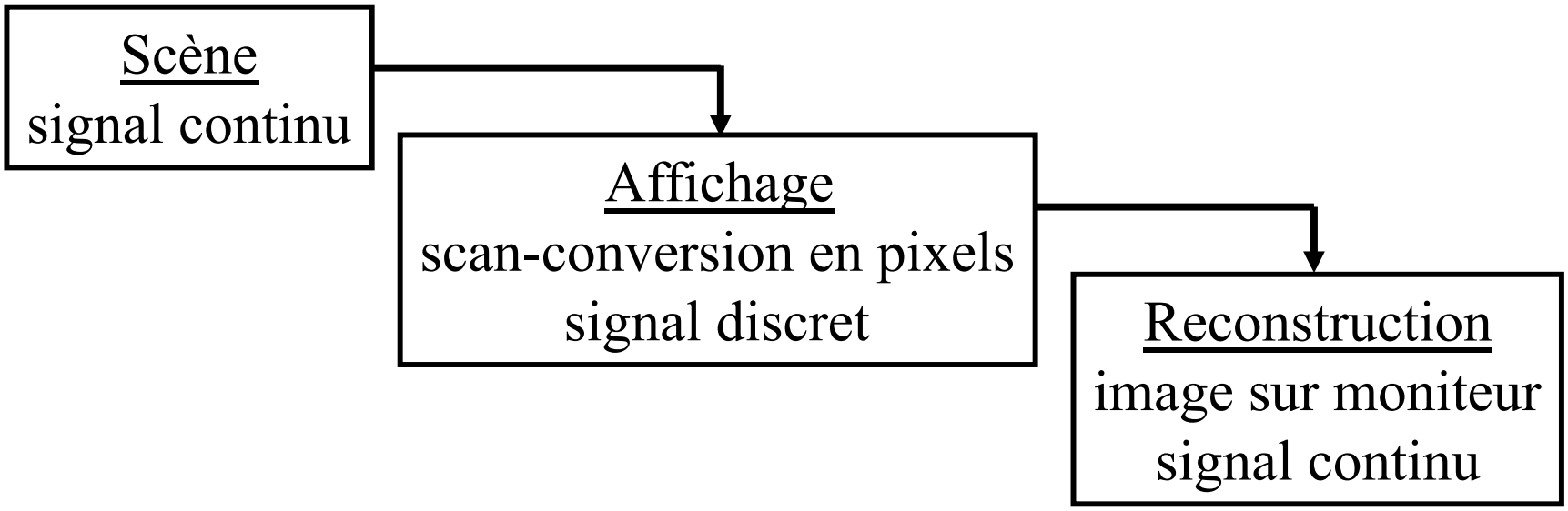


Signal continu et signal discret

- Une scène projetée est un signal continu
  - Il existe une valeur pour chaque point de l'image
  - Les valeurs varient arbitrairement d'un point au suivant
  - De hautes fréquences apparaissent aux silhouettes, dans les textures, sur l'illumination
- Le balayage (*scan-conversion*) convertit le signal continu de la scène en un signal discret de l'image
  - On prend souvent la valeur au centre du pixel
  - Les variations de valeurs deviennent limitées

Discrétisation

- En passant d'un signal continu à un signal discret, on perd de l'information
- Le filtrage essaie de choisir les meilleures valeurs pour le signal discret afin de mieux approximer le signal continu original



Echantillonnage ponctuel

Un seul échantillon au centre de chaque pixel crée de l'aliassage

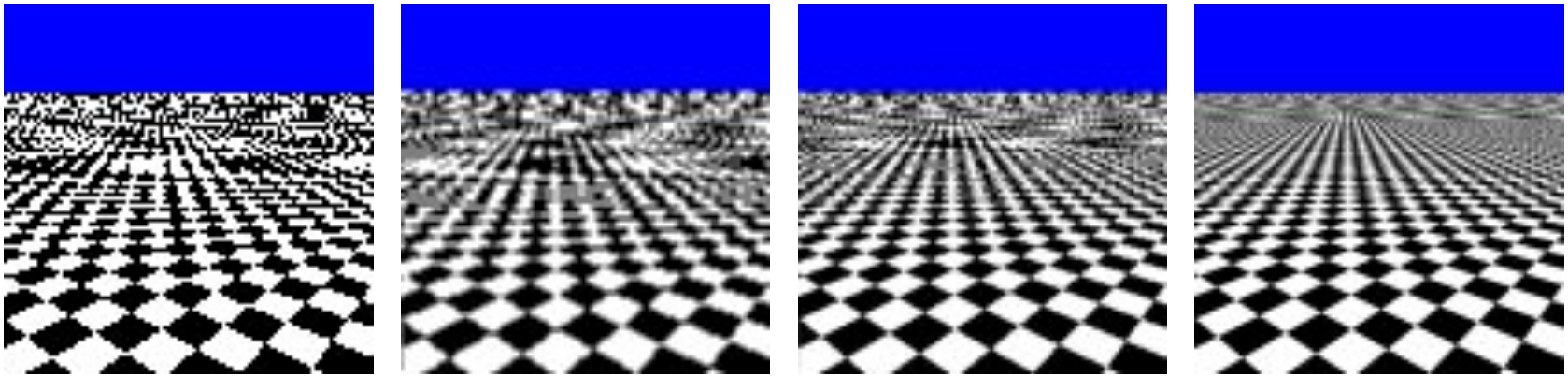
Si on utilise plus d'échantillons, on a

- plus d'information pour mieux représenter le pixel
- des petits sous-pixels pour reconstruire le pixel

ce qui résulte potentiellement en moins d'aliassage

Sur-échantillonnage

On peut échantillonner régulièrement dans un pixel et pondéré également (filtre boîte) les couleurs des échantillons

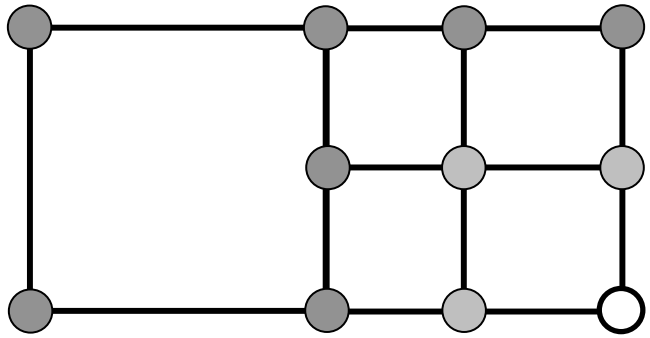


1 par pixel | 2 par pixel | 4 par pixel | 16 par pixel

Par expérience, un sur-échantillonnage de 4x4 est souvent considéré comme un bon compromis qualité - temps de calcul

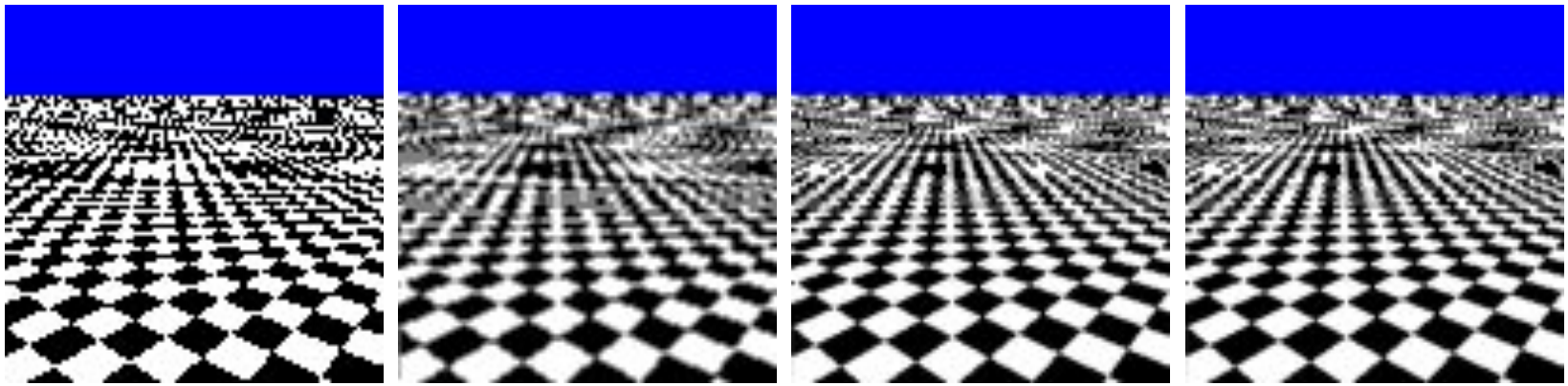
Echantillonnage adaptatif

On peut aussi subdiviser régulièrement mais adaptativement selon les différences de couleur des échantillons adjacents



Echantillonnage adaptatif

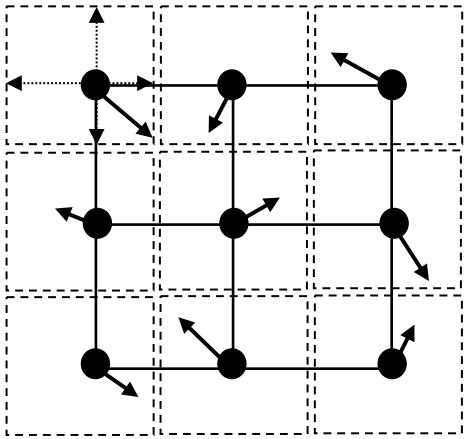
Les nombres minimum et maximum de niveaux de subdivision sont indiqués sous chaque image

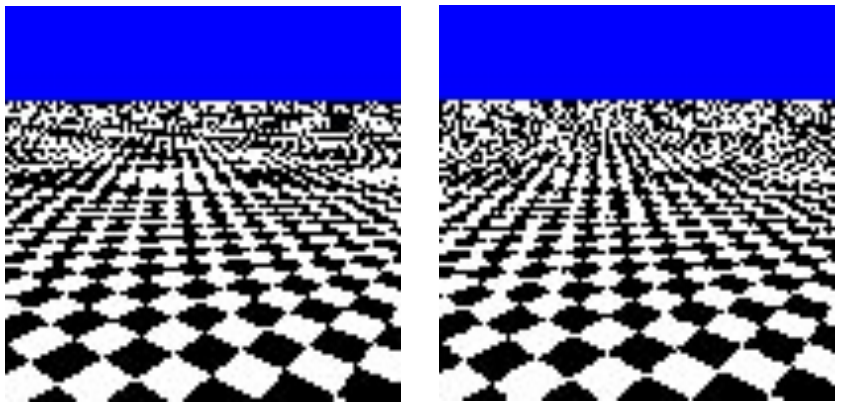


min 0 max 0	min 1 max 2	min 1 max 4	min 2 max 4

Perturbations stochastiques *(jittering)*

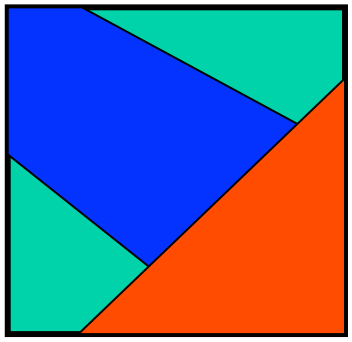
- Au lieu d'échantillonner régulièrement, on peut perturber aléatoirement (*jitter*) la position de chaque échantillon
- On introduit alors du bruit dans l'image qui remplace en partie l'aliassage

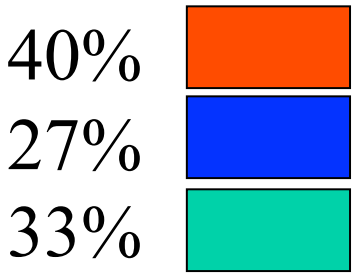


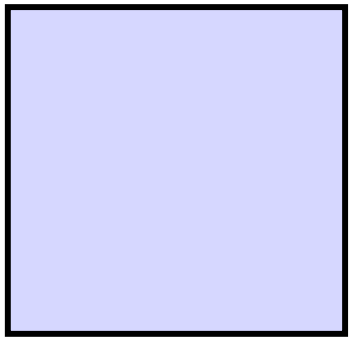


Echantillonnage d'aire

- La contribution d'une surface dépend de la partie du pixel recouverte par cette surface.
- On intègre la surface dans le domaine (aire) du pixel en divisant par l'aire du pixel
- Ceci correspond à appliquer un filtre boîte sur le pixel
- Un filtre boîte réduit l'aliassage mais en l'élimine pas (analyse de Fourier dans l'espace fréquentiel)

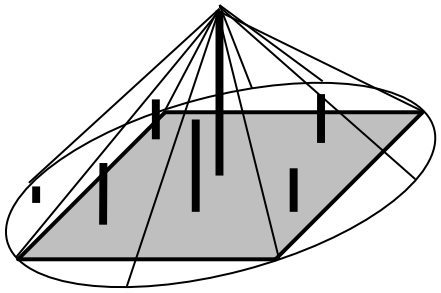




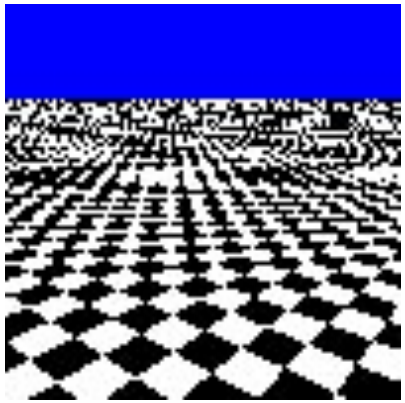


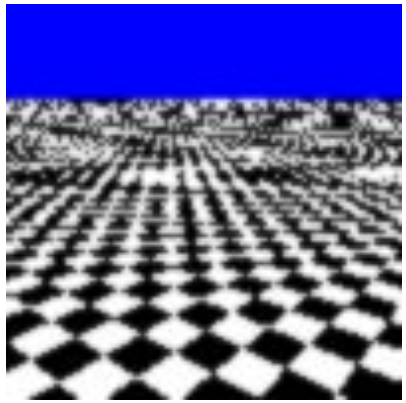
Filtrage de pixels adjacents

- Si la contribution ne dépend pas de la position à l'intérieur du pixel, il s'agit d'une somme non-pondérée et d'un filtre boîte
- Au lieu de faire la moyenne des couleurs des échantillons, on peut leur donner un poids différent en fonction de leur distance du centre du pixel



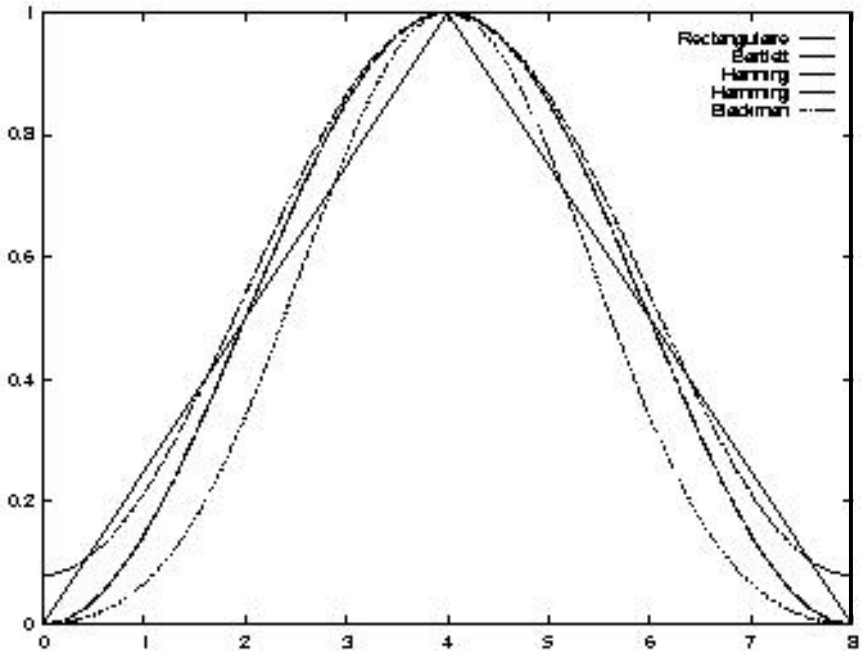
filtre linéaire à
base circulaire





Filtres

- La pondération dépend de la forme du filtre

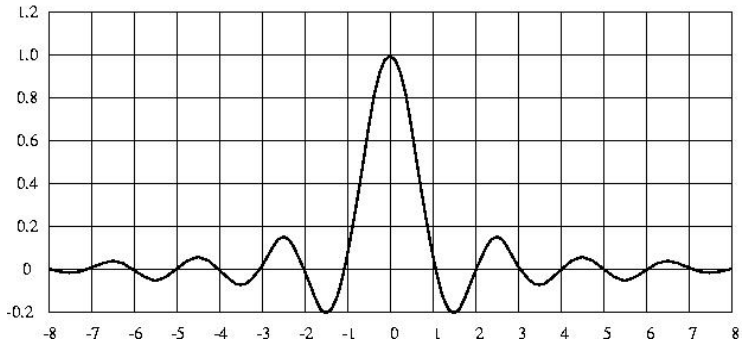


- Un filtre s'étend souvent sur plus d'un pixel

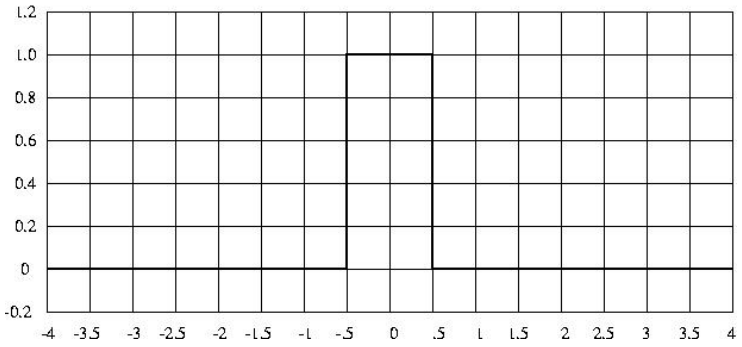
Filtre idéal

- Le filtre idéal est appelé le $\text{sinc} = \dfrac{\sin(\pi\ x)}{\pi\ x}$

- Le filtre a la forme suivante



domaine spatial



domaine fréquentiel

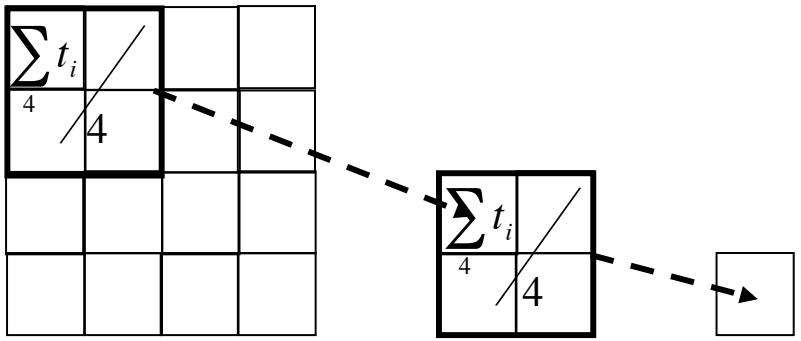
- Malheureusement ce filtre a un support infini

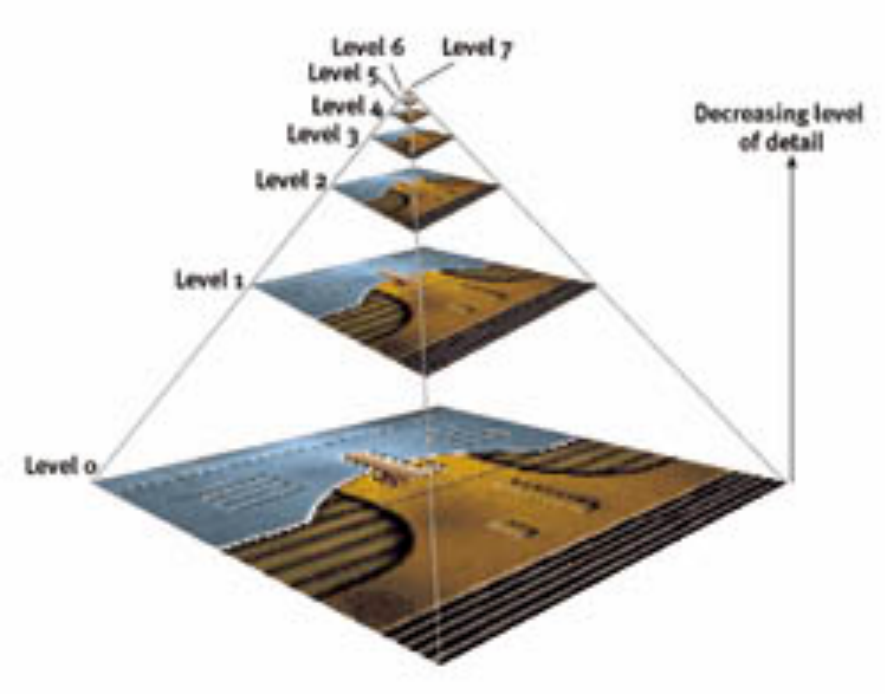
Préfiltrage

- Il est parfois avantageux de calculer au préalable des valeurs filtrées des textures avant le processus de rendu
  - texture est répétée sur une ou plusieurs surfaces
  - texture est utilisée dans plusieurs images d'une animation
- On risque alors de calculer des valeurs inutilement, mais en général, préfiltrer est avantageux
- Si on incorpore des texels hors du pixel, on substitue du flou au bruit, ce qui est plus désirable

MIP mapping (multum in parvo)

- Les éléments de la texture sont préfiltrés (moyennés) dans une pyramide

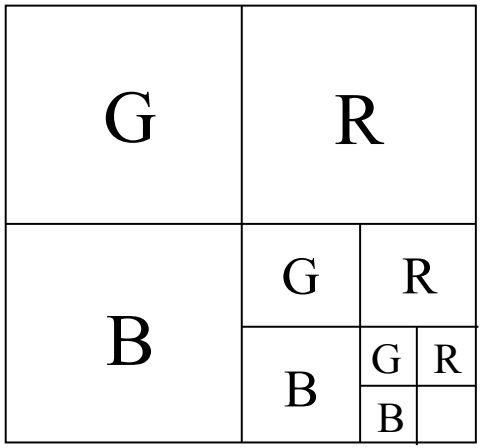




Ertoz

MIP mapping (multum in parvo)

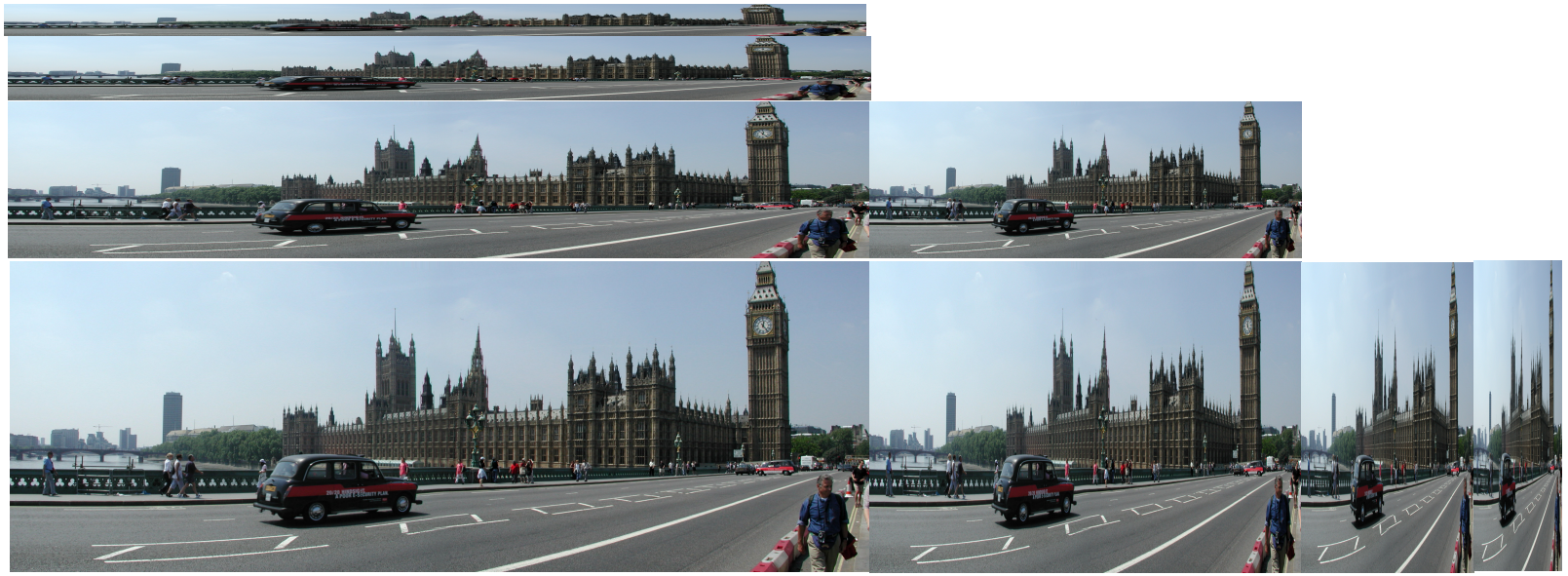
- La texture résulte en 4/3 de sa taille originale
- Lorsque plusieurs texels projettent dans un pixel, le niveau le plus approprié dans la pyramide est choisi
- En hardware sur plusieurs cartes



$$512 \times 512 \times 24 = 1024 \times 1024 \times 8$$

RIP map

[Haines-Moeller]



Construction d'une table de sommation

- En mipmap, forme du filtre est carrée
- Un filtre de forme rectangulaire pourrait être souvent plus approprié (quoiqu'aligné sur les axes)

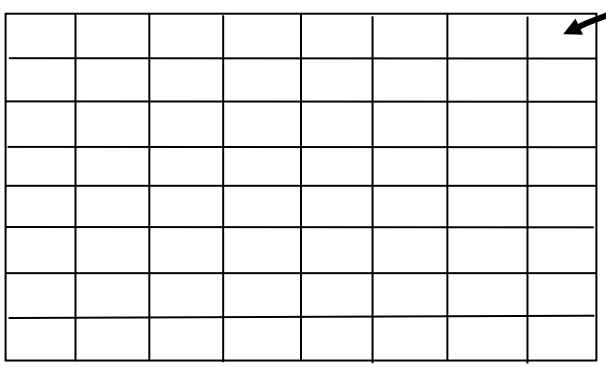
Texture

1	1	1	1
1	1	1	1
1	1	1	1
1	1	1	1

Table de sommation

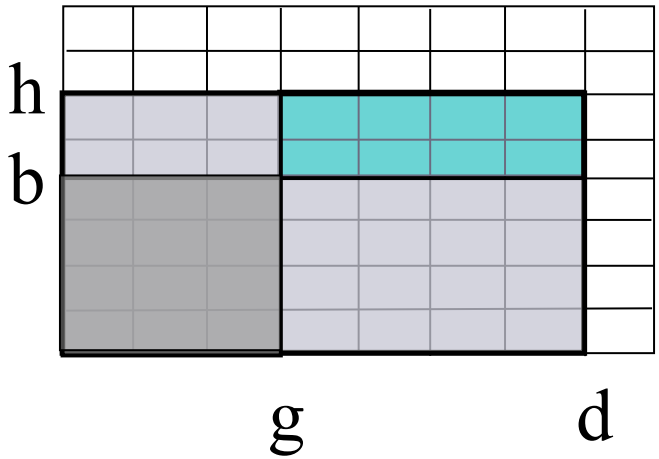
4	8	12	16
3	6	9	12
2	4	6	8
1	2	3	4

Table de sommation



$$\sum C_{ij}$$

$2^r \times 2^r \times b$ bits : $2r + b$ bits par pixel



$$C = \frac{C_{dh} - C_{db} - C_{gh} + C_{gb}}{\text{aire du rectangle}}$$

Aucun filtrage

Mip map

Table de sommation