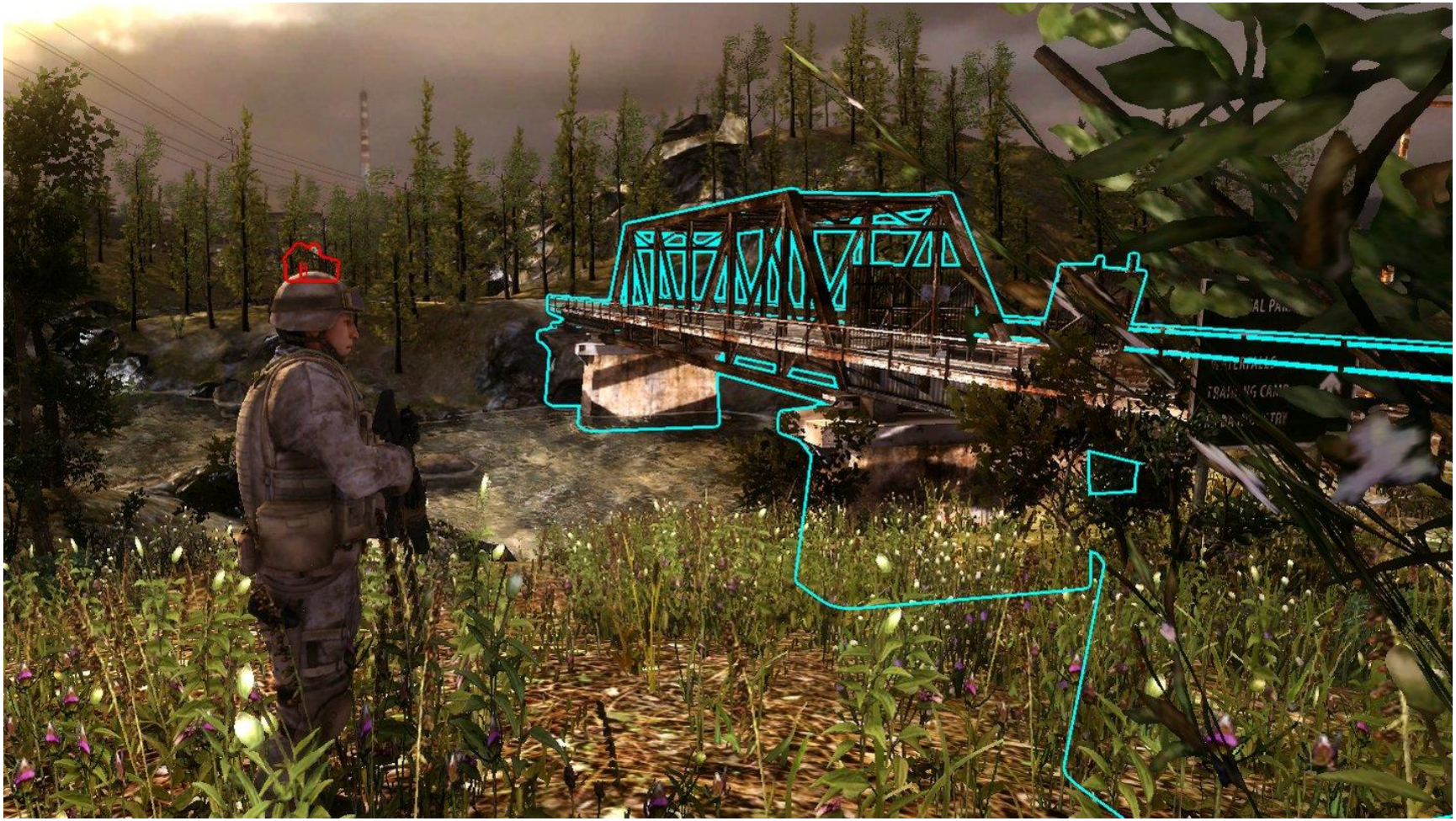
Shading Avancé

Cel shading, outline, DOF, god rays, bloom, Motion Blur, fisheye, vignetting, SSAO, Shadow Maps, filmic tonemaping, HDR

Toon Shading

```
uniform vec3 lightDir;
varying vec3 normal;
void main()
{
        float intensity;
        vec4 color;
        intensity = dot(lightDir,normal);
        if (intensity > 0.95)
        color = vec4(1.0,0.5,0.5,1.0);
        else if (intensity > 0.5)
        color = vec4(0.6,0.3,0.3,1.0);
        else if (intensity > 0.25)
        color = vec4(0.4,0.2,0.2,1.0);
        else
        color = vec4(0.2,0.1,0.1,1.0);
        gl_FragColor = color;
}
```

Outline



Outline

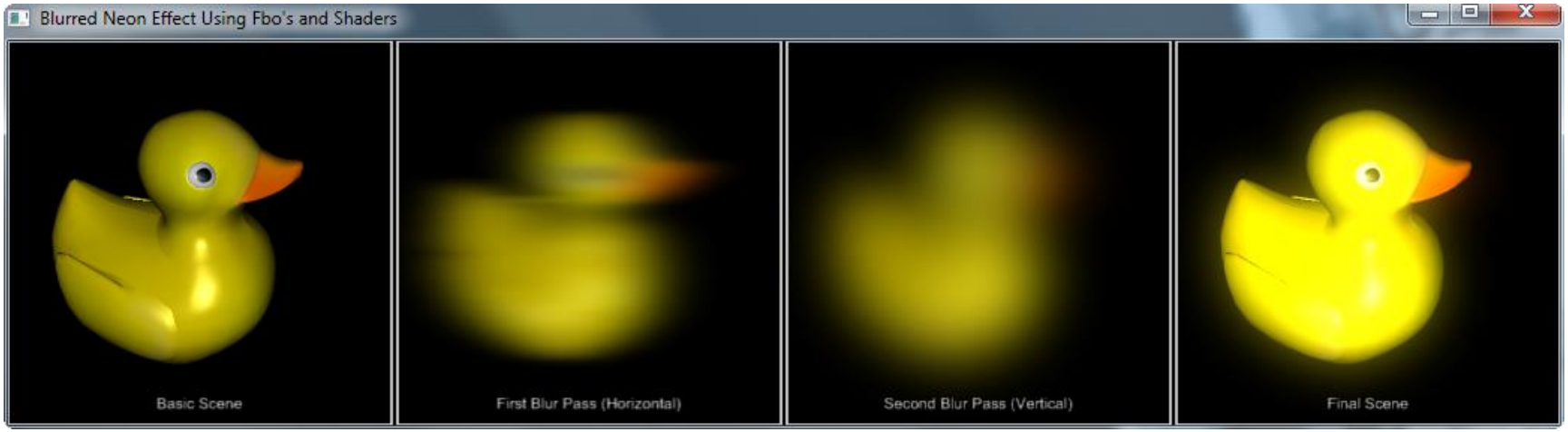
- Tech 1
  - 1) rendu des faces arrières de l'objet, avec un déplacement des sommets sur la normale, de la couleur de l'outline
  - 2) rendre l'objet normalement
  - 3) avec un blur, on a un glow
  - 4) une passe, risque de Z fight
- Tech 2
  - Sobel Filter (passe haut) en post process

Bloom

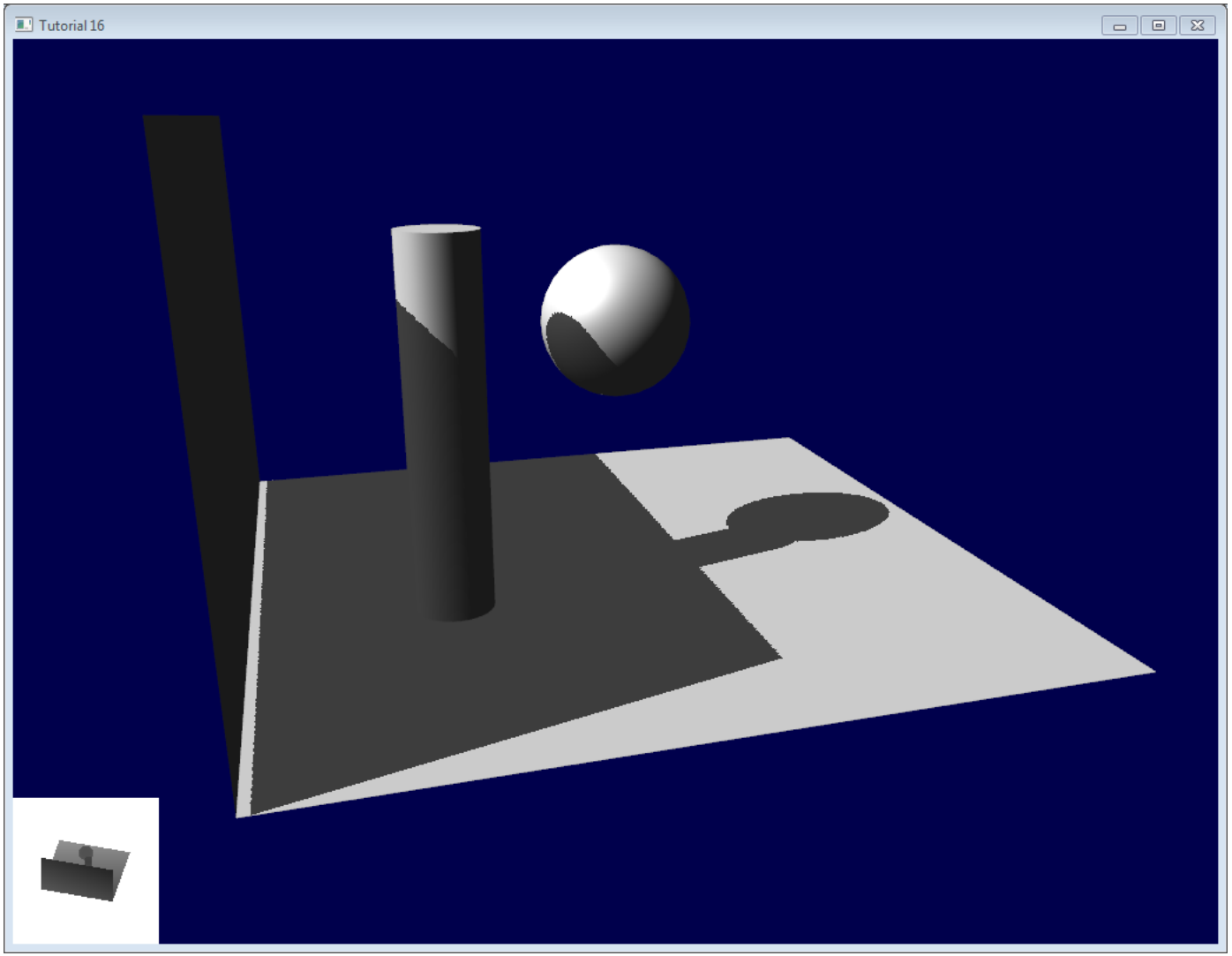
Bloom

- Rendre toutes les parties lumineuses dans un buffer
- Filtre de flou sur ce buffer
- Mélange au buffer principal

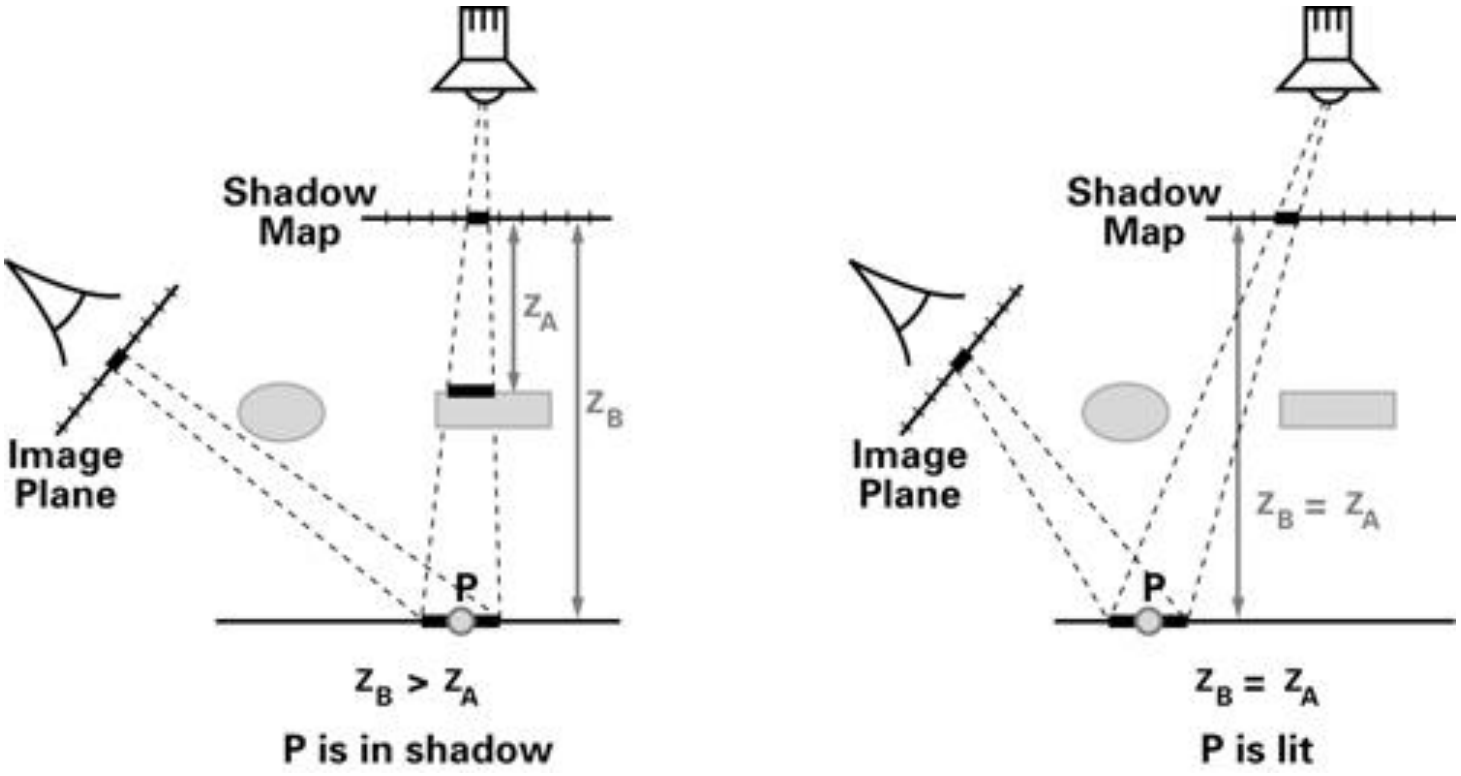
Bloom


Blurred Neon Effect Using Fbo's and Shaders
Basic Scene
First Blur Pass (Horizontal)
Second Blur Pass (Vertical)
Final Scene

Shadow Maps

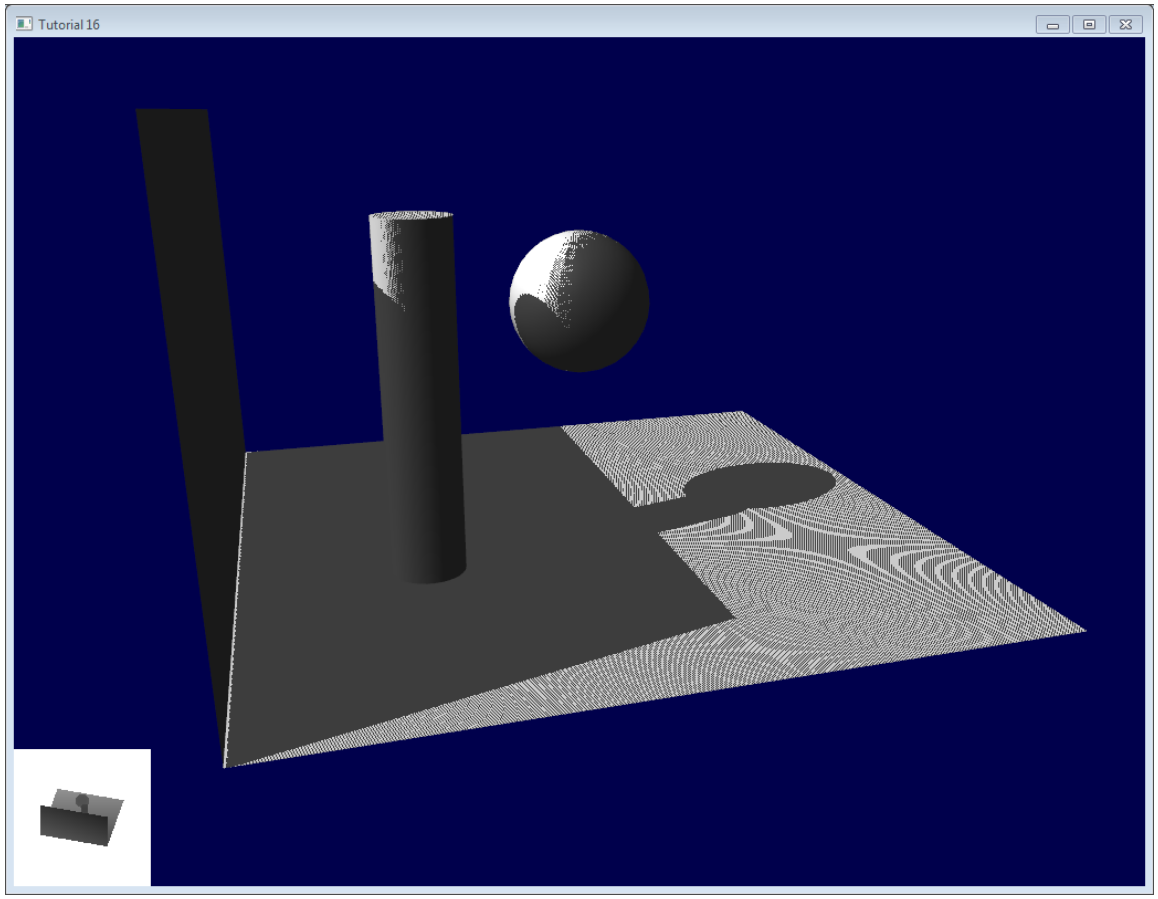


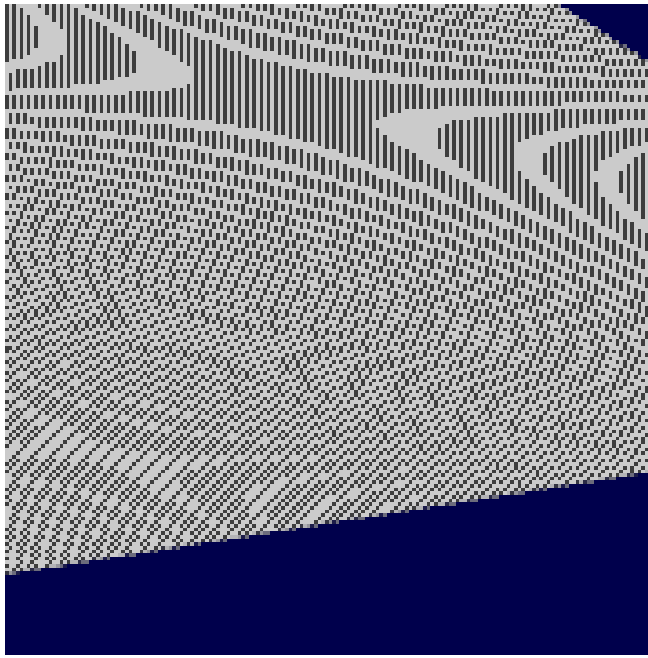
Shadow Maps


Shadow Map
Image Plane
Z_A
Z_B
P
$Z_B > Z_A$
P is in shadow
Shadow Map
Image Plane
$Z_B = Z_A$
P
$Z_B = Z_A$
P is lit

Shadow Maps

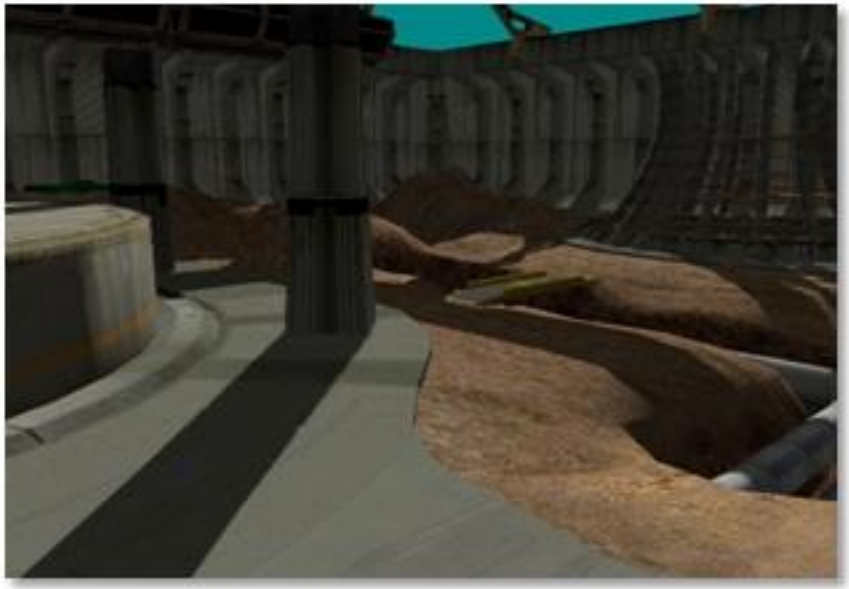
- Pb d’acne (arrondi shadow map)





Shadow Maps

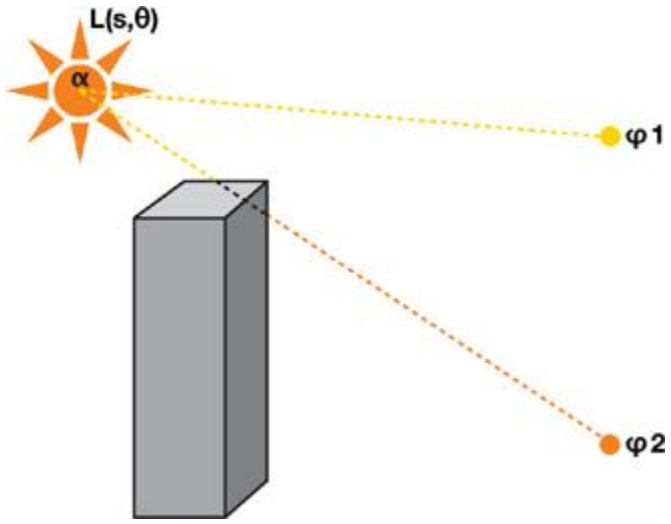
- Une solution acne :un bias (faire une comparaison avec une marge plus importante que juste !=)
- La bias agrave le peter panning



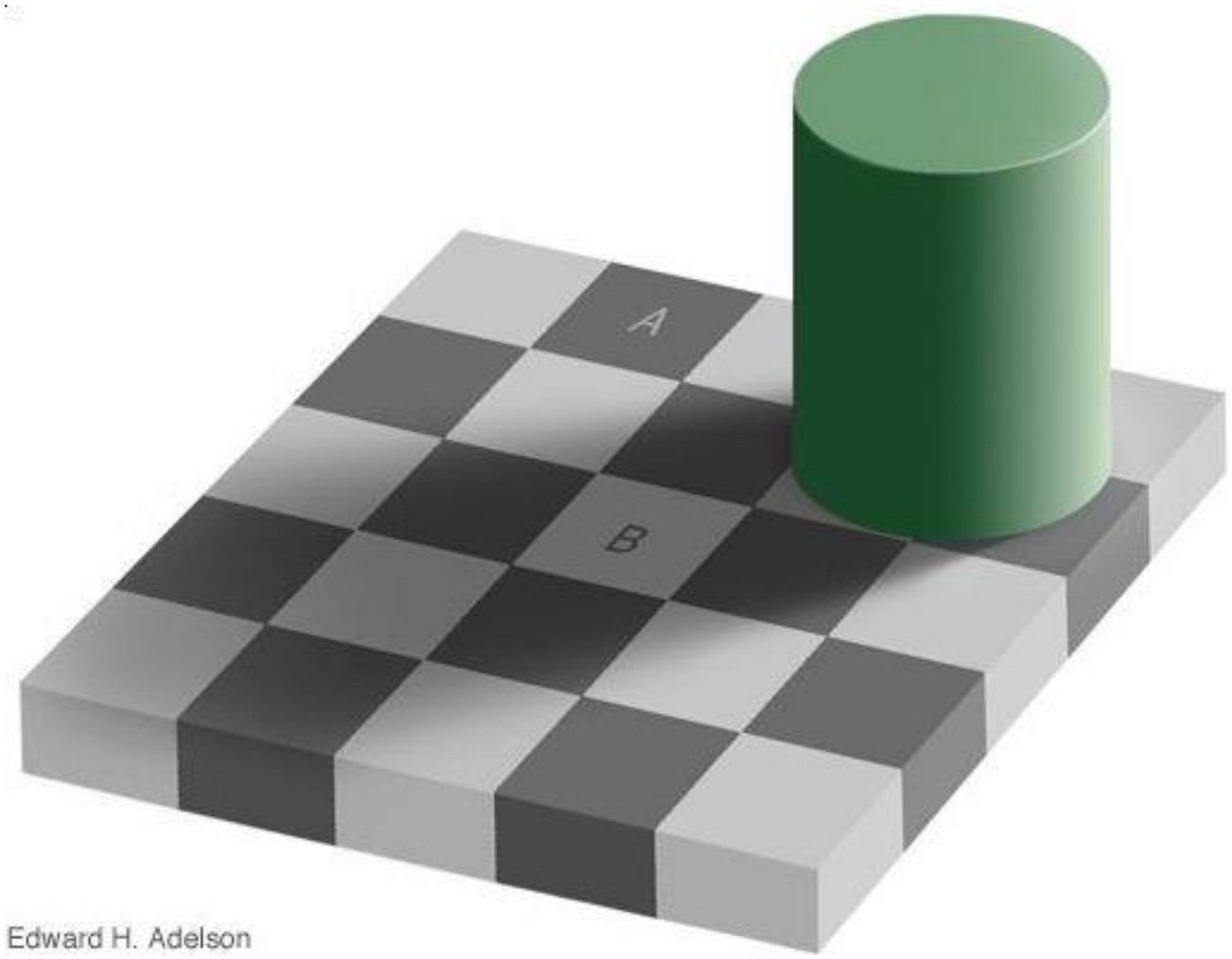
God Rays

God Rays / Sun Shaft

- http://http.developer.nvidia.com/GPUGems3/gpugems3_ch13.html

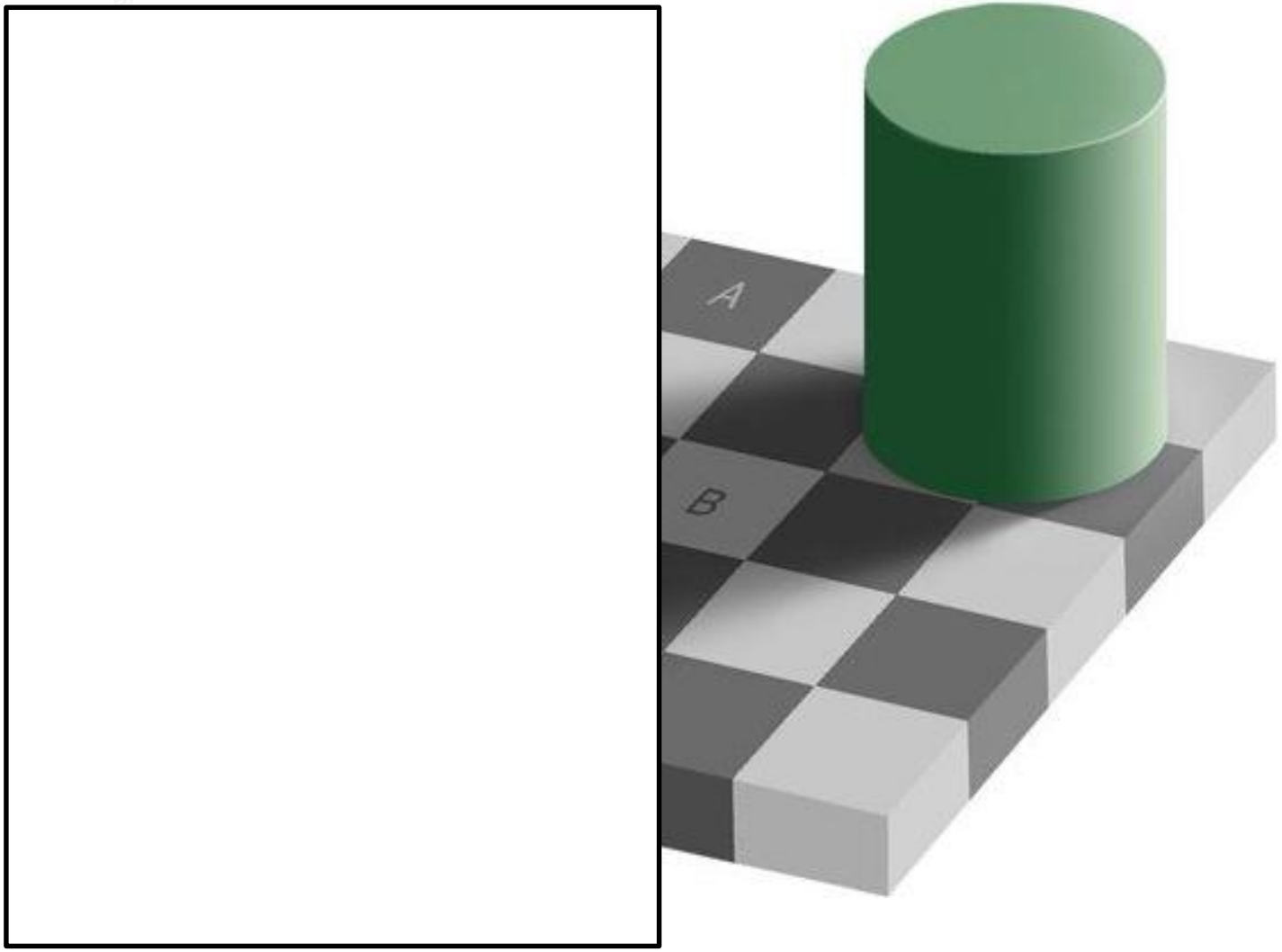


HDR Tonemaping

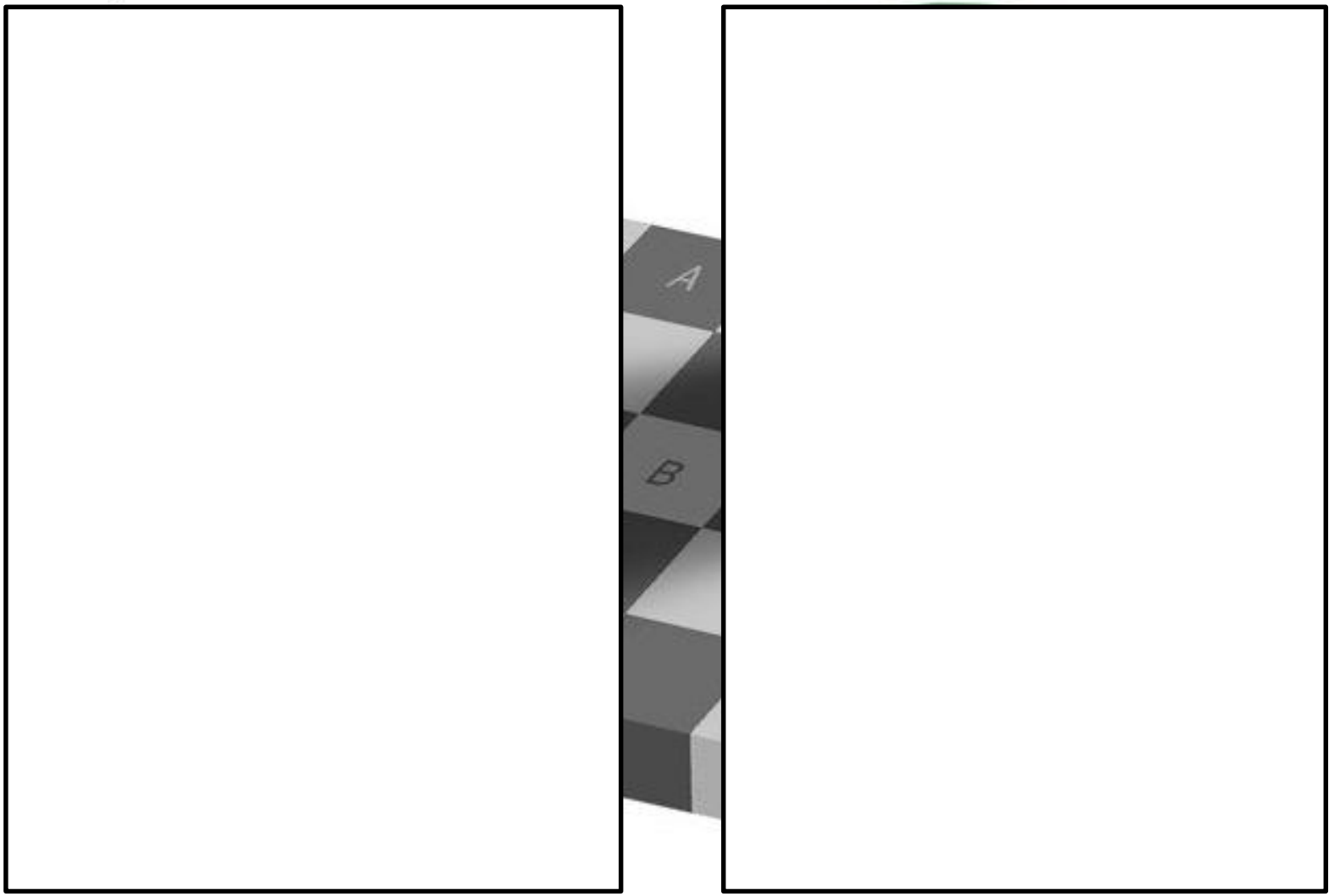


Edward H. Adelson

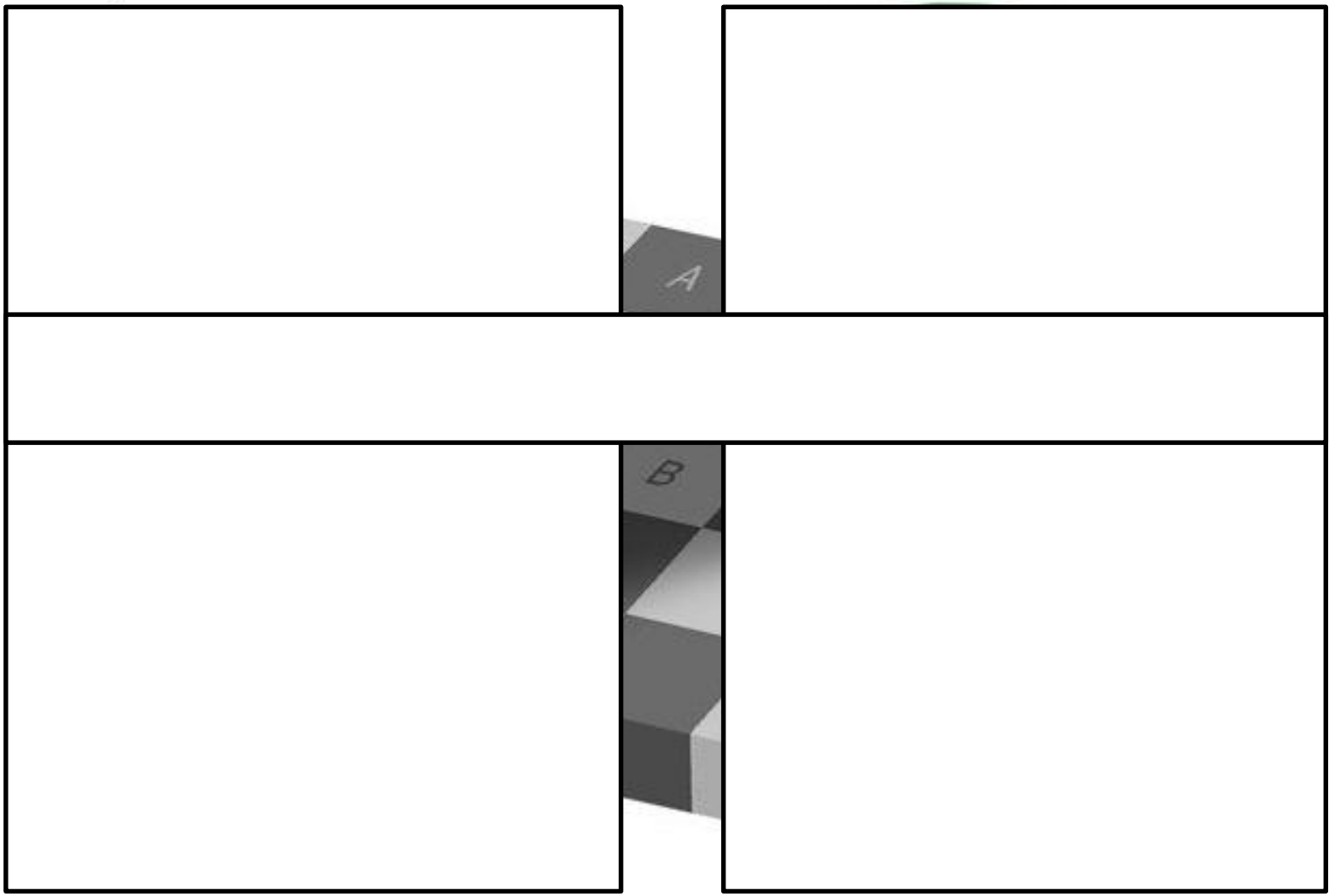
HDR Tonemaping



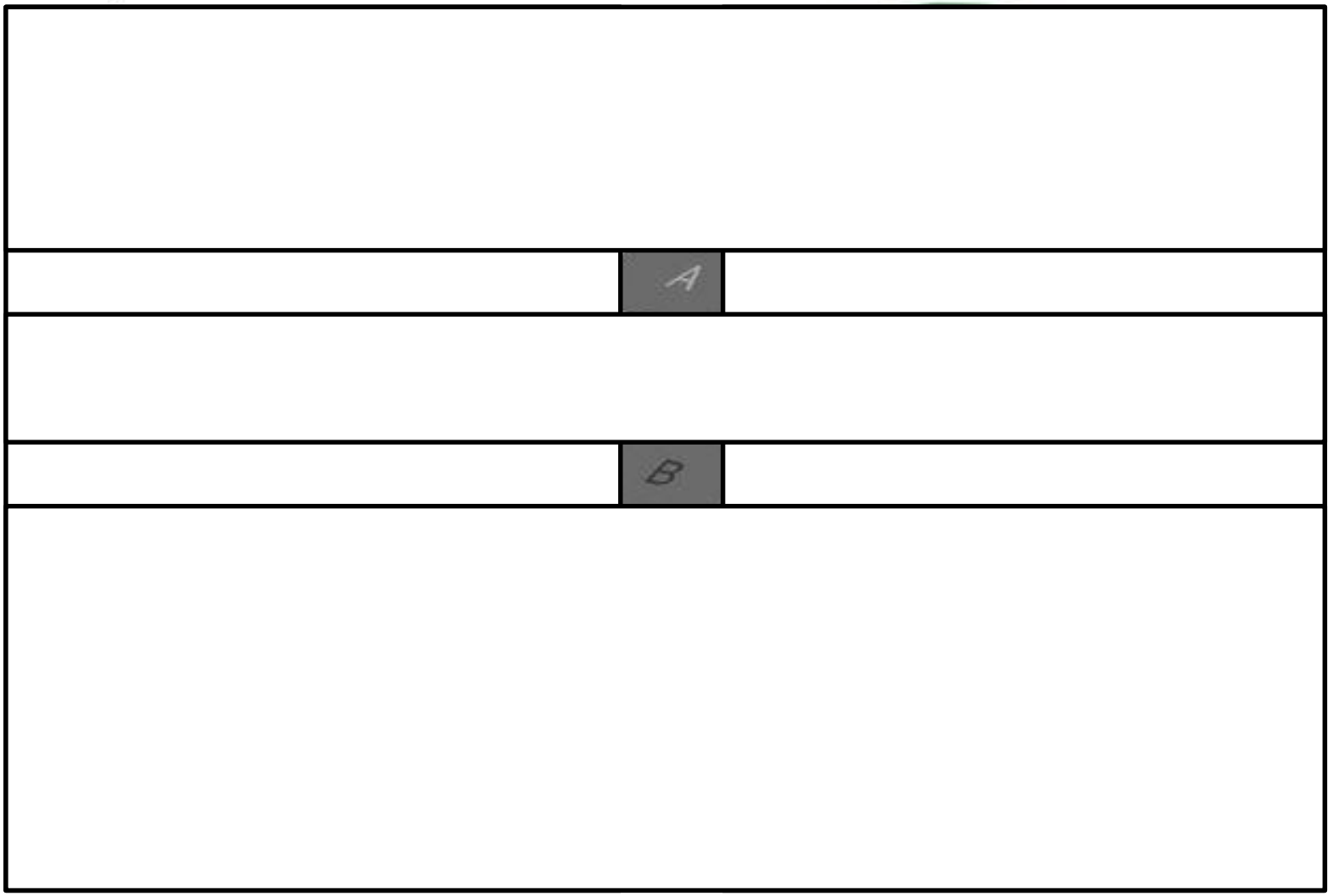
HDR Tonemaping



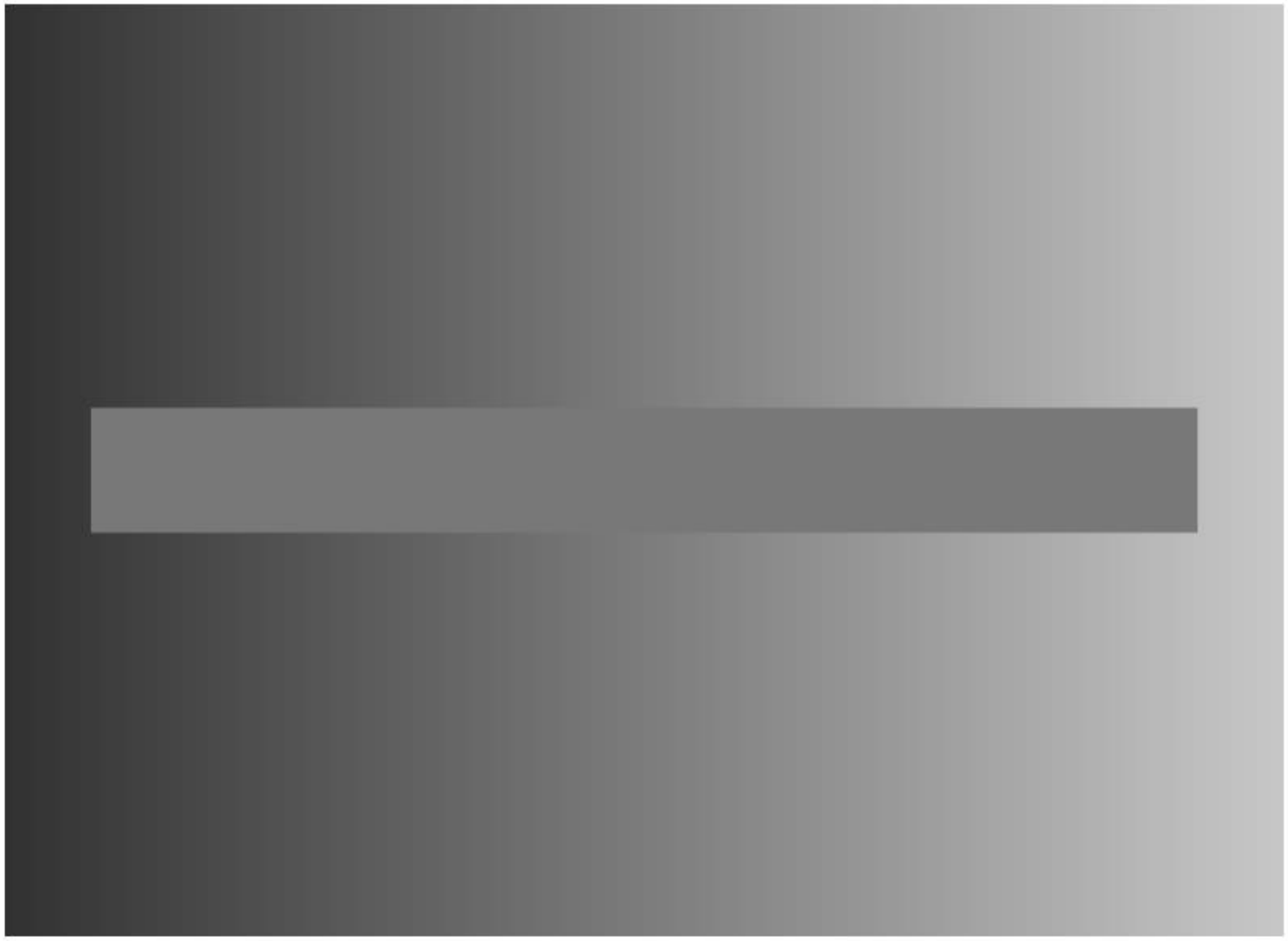
HDR Tonemaping



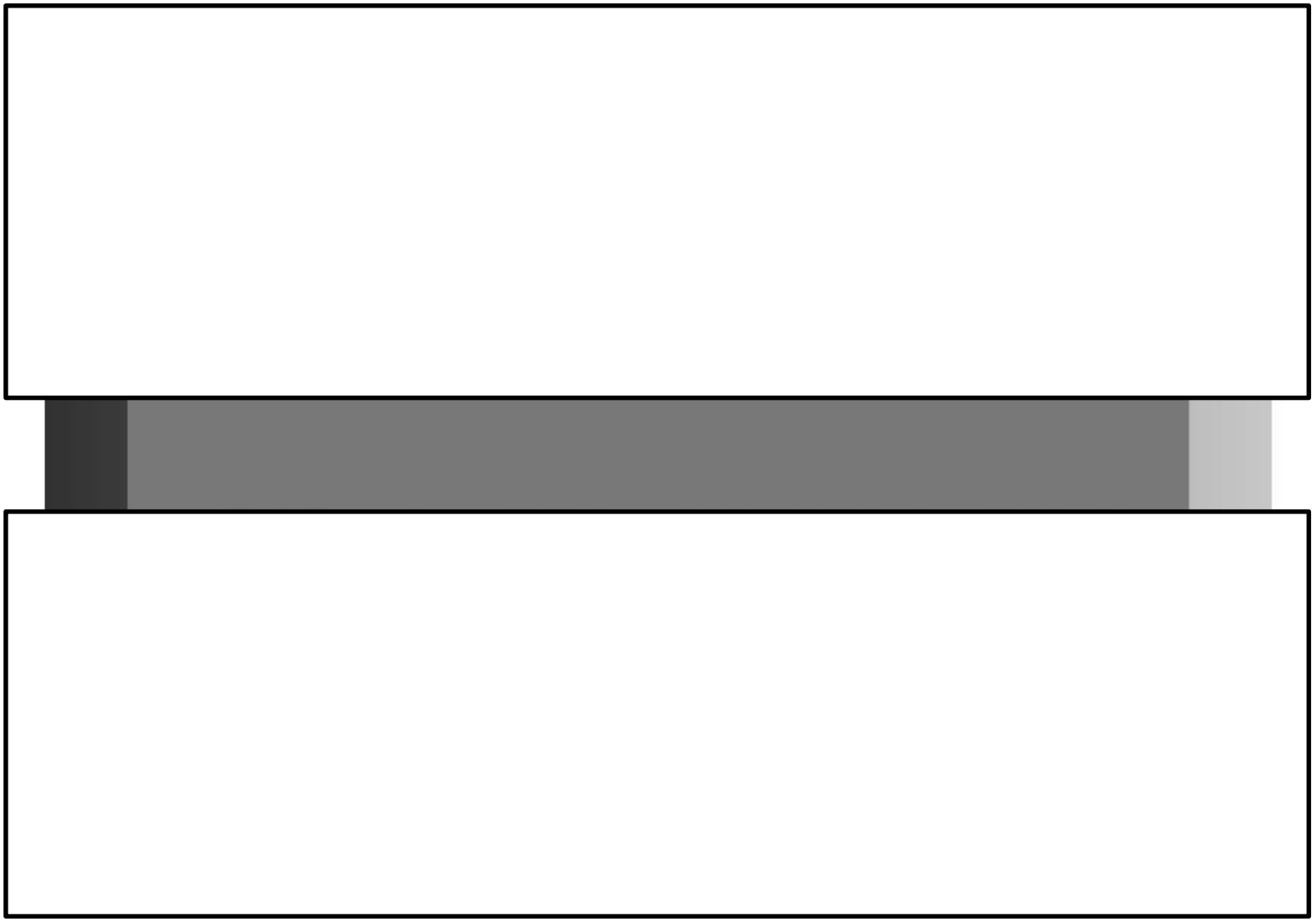
HDR Tonemaping



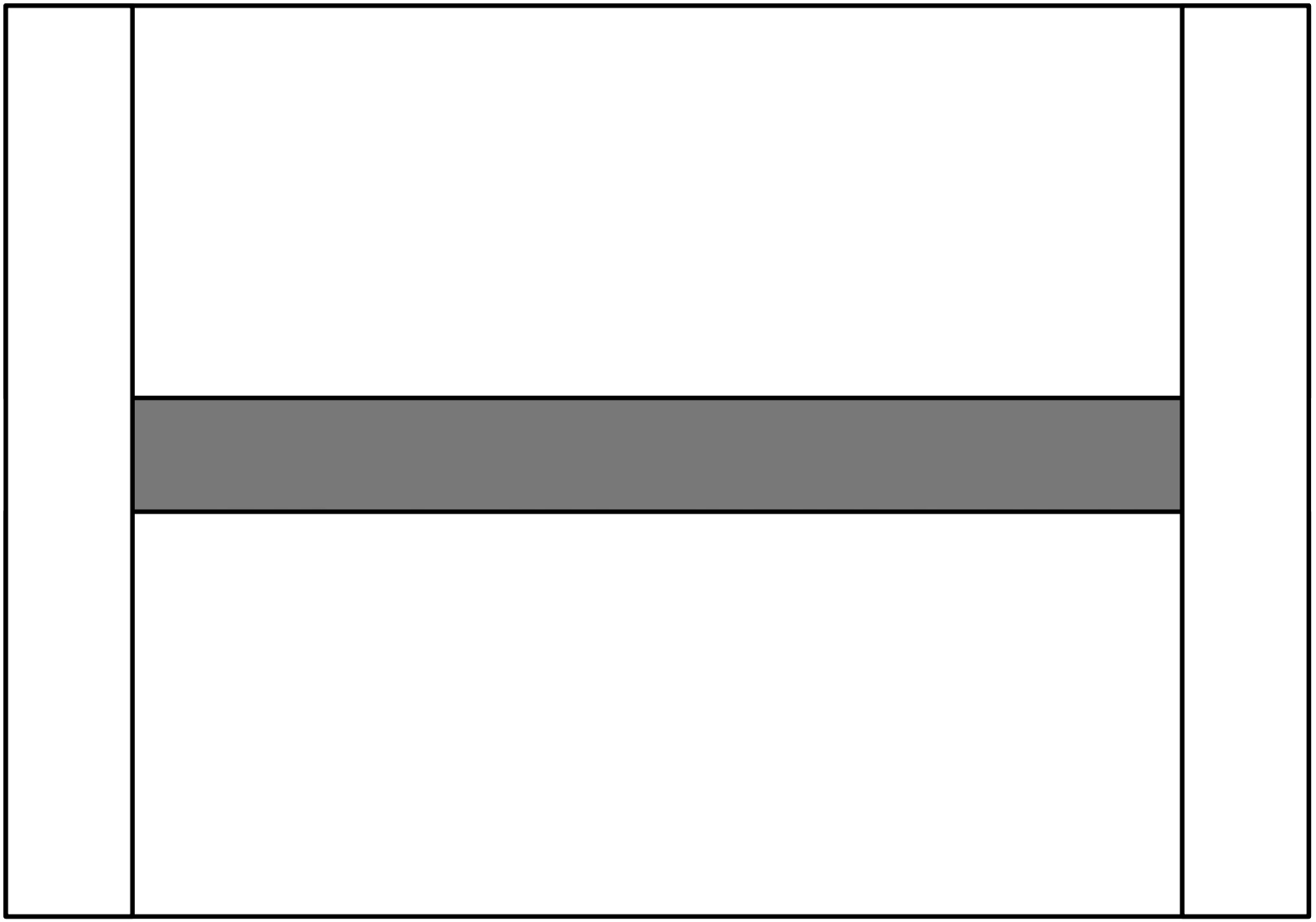
HDR Tonemaping



HDR Tonemaping



HDR Tonemaping



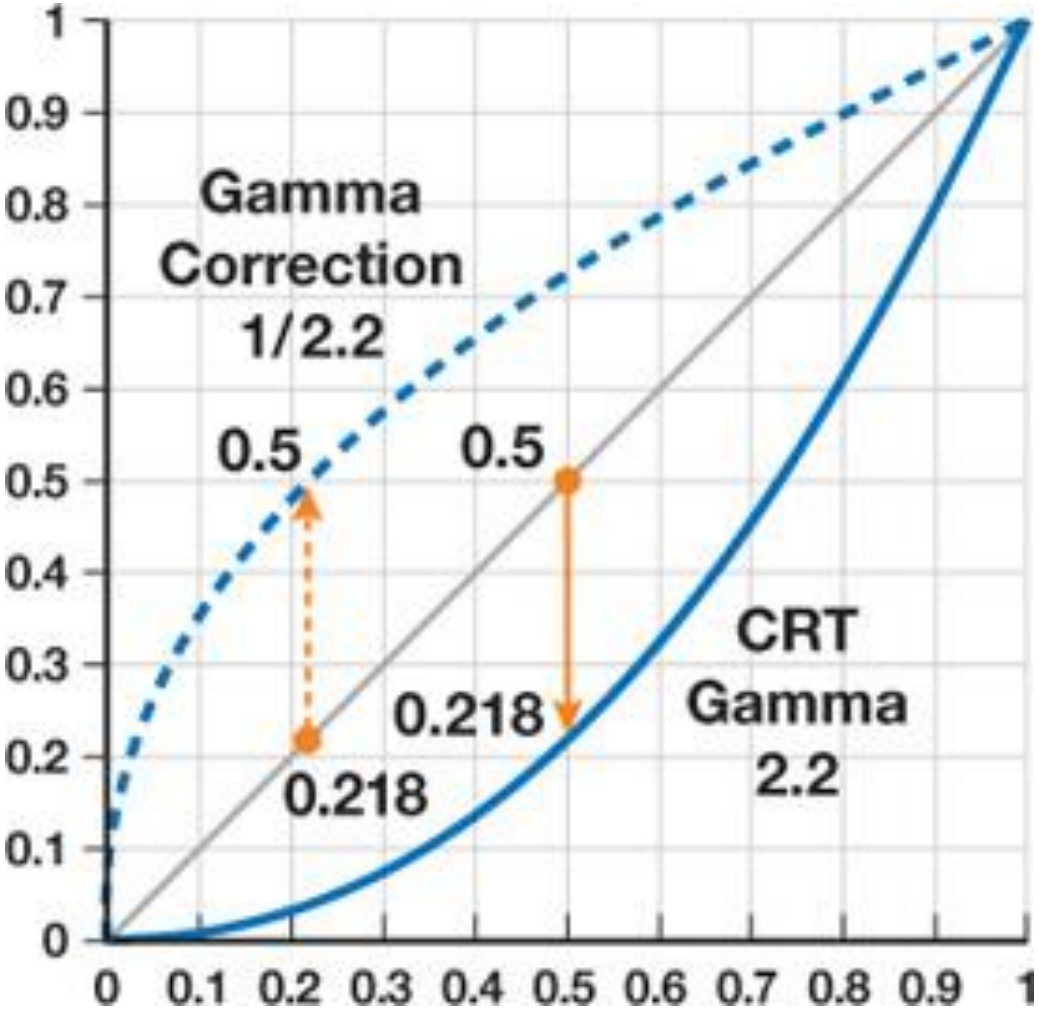
HDR & tone mapping

HDR & tone mapping

HDR tonemapping

- Rendu dans un buffer intermédiaire avec une résolutions supérieure à celle de l'image finale
- Jouer ensuite avec le mapping des zones sous exposées et sur exposées
- Modifier l'exposition de base pour simuler l'iris (sortie de tunnel sombre) : Automatic Exposure Adjustment

Correction Gamma



Correction Gamma

(a) (b)

Correction Gamma

(a) (b)

Linear / Gamma space

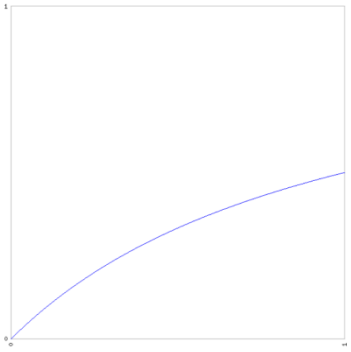
- Les textures sont faites par des artistes. Donc sur des écrans, avec la correction gamma. Il faut donc corriger en entrée
  - Color = pow(tex2D(Sampler,Uv),2.2);
- On calcule ensuite tout le lighting dans l'espace linéaire
- Au moment du rendu, on applique a nouveau la correction gamma
  - finalCol = float4(pow(finalCol.rgb, 1.0 / 2.2), finalCol.a);

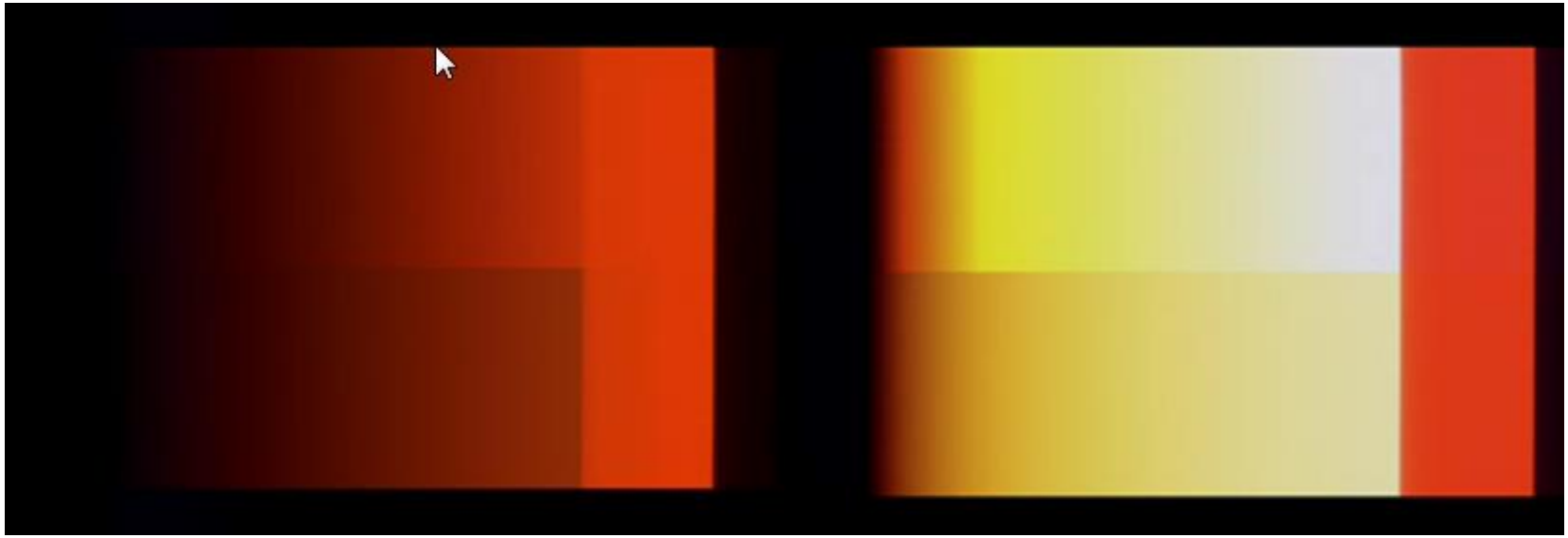
Linear / Gamma space

- Peut etre automatique avec opengl et dx
  - glEnable(GL_FRAMEBUFFER_SRGB) et format de texture GL_SRGB
  - Extentions GL_EXT_texture_sRGB and GL_EXT_framebuffer_sRGB

Reinhard tonemaping

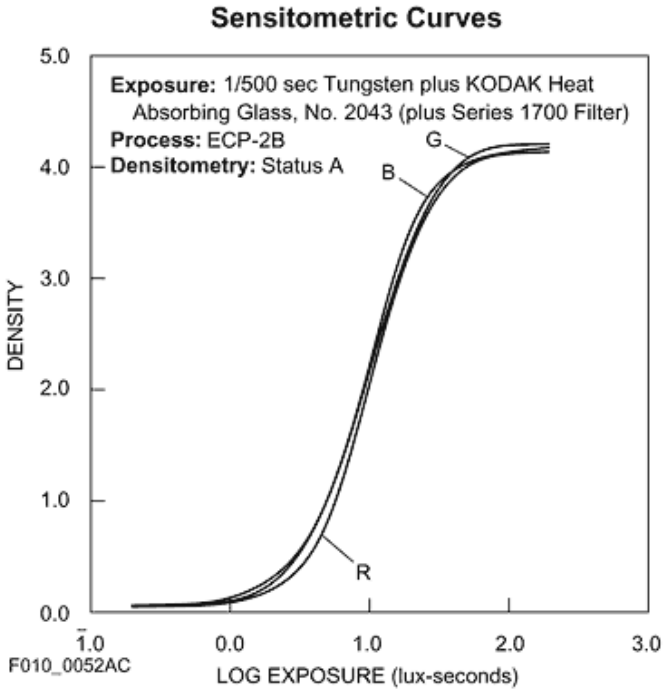
- Color = color / (1+color)
- Désature globalement l’image, mais améliore les zones surexposées.





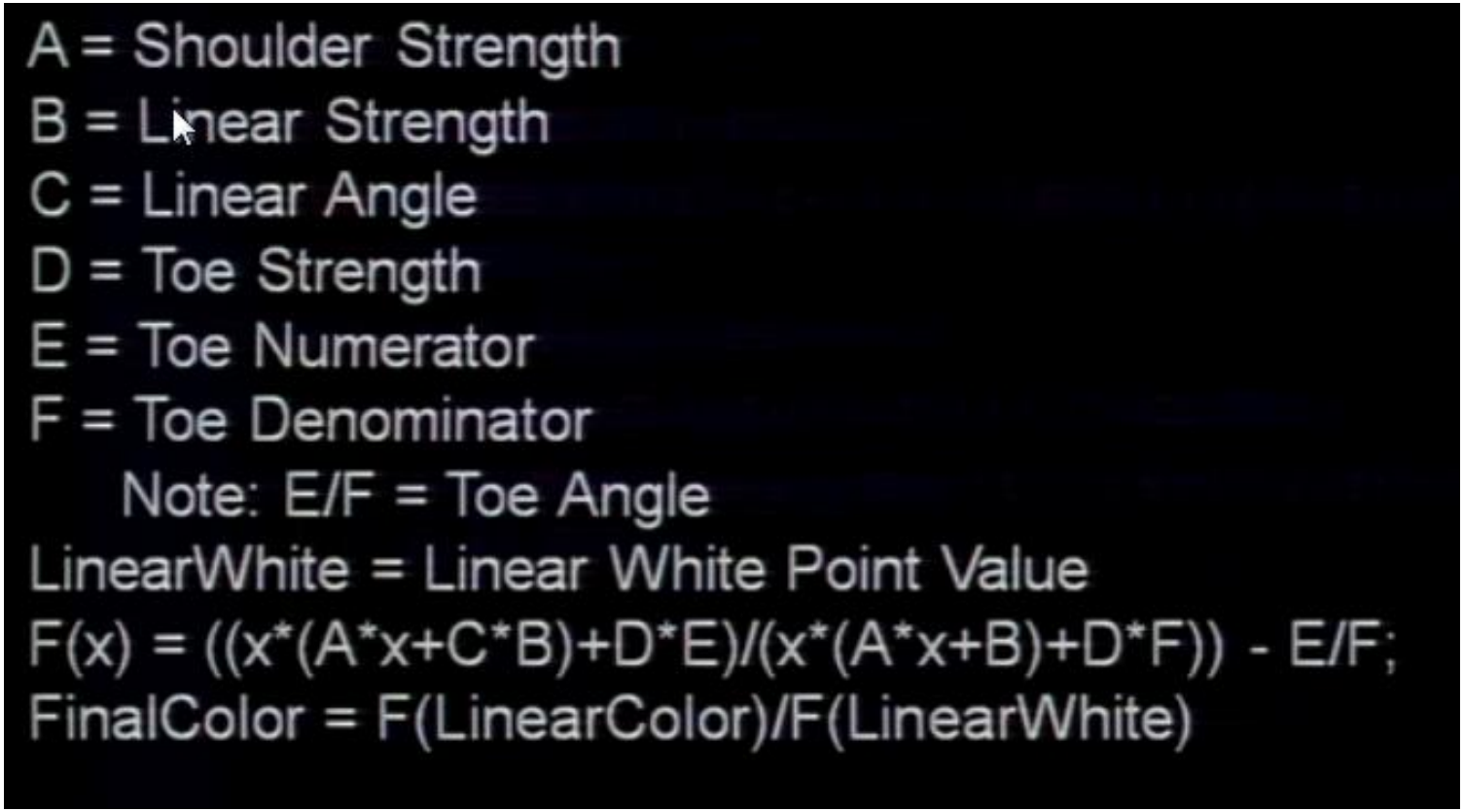
Filmic tonemaping

- Utiliser la courbe d'une pellicule. (voir gdc2010 john hable / naughty dog / uncharted 2)
- Utiliser une texture pour la courbe



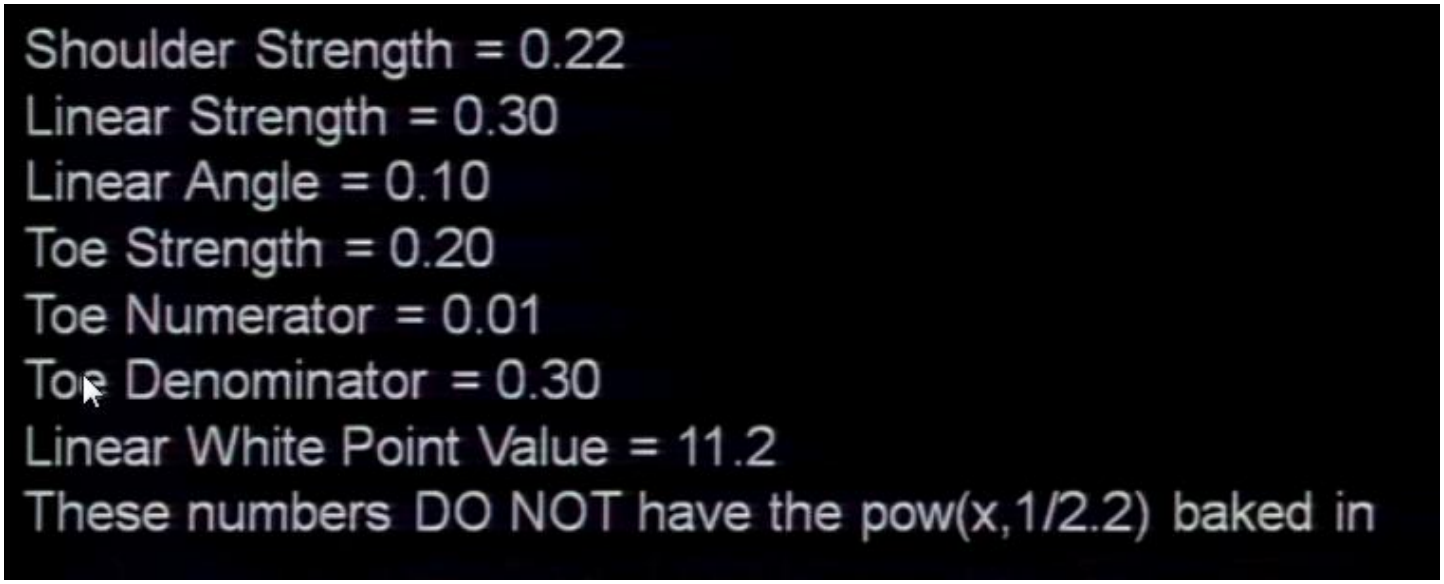
Filmic tonemaping

- Utiliser une texture pour la courbe
- Ou fonction maison (uncharted 2) :



Filmic tonemaping

- Default pour uncharted 2



Espace Colorimétrique

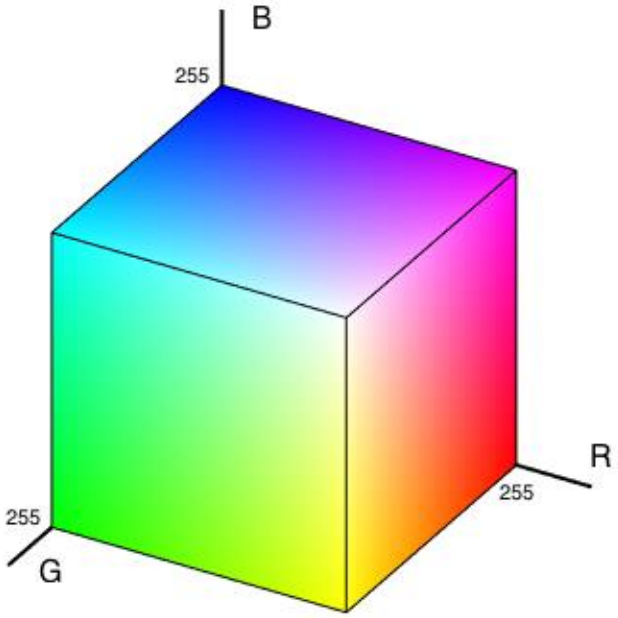
- Exemple : génération d’un visage coloré a partir de niveaux de gris (ma tête dans la photocopieuse)

Espace Colorimétrique

- Utiliser une couleur aléatoire « sombre » pour le fond, et des couleurs complémentaires « lumineuses » pour le visage
- Détail : le niveau de gris donne a peu près une profondeur, l'utiliser pour donner un effet de texture mapping et garder le volume (plutôt que des transitions horizontales)

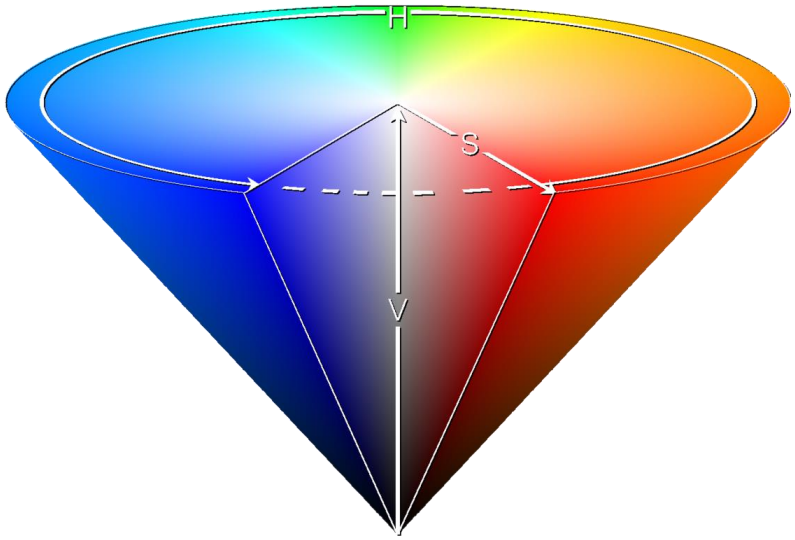
Espace RVB

- Travailler en RVB ?
- Pb: ne faire varier que la teinte, garder une saturation et une brillance constante…



Espace HSV

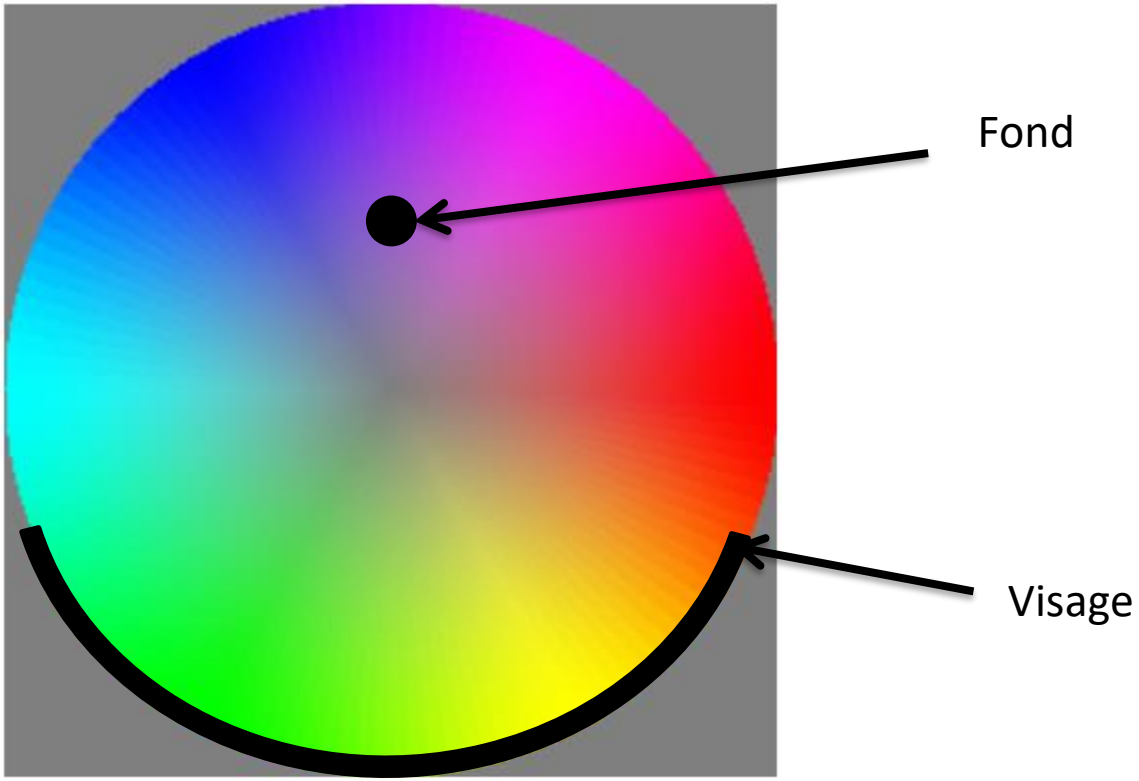
- HSV plus adapté :
  - H : teinte de la couleur (hue)
  - S : saturation ou pureté (saturation à 0 = niveaux de gris)
  - V : brillance (value), ou luminosité.

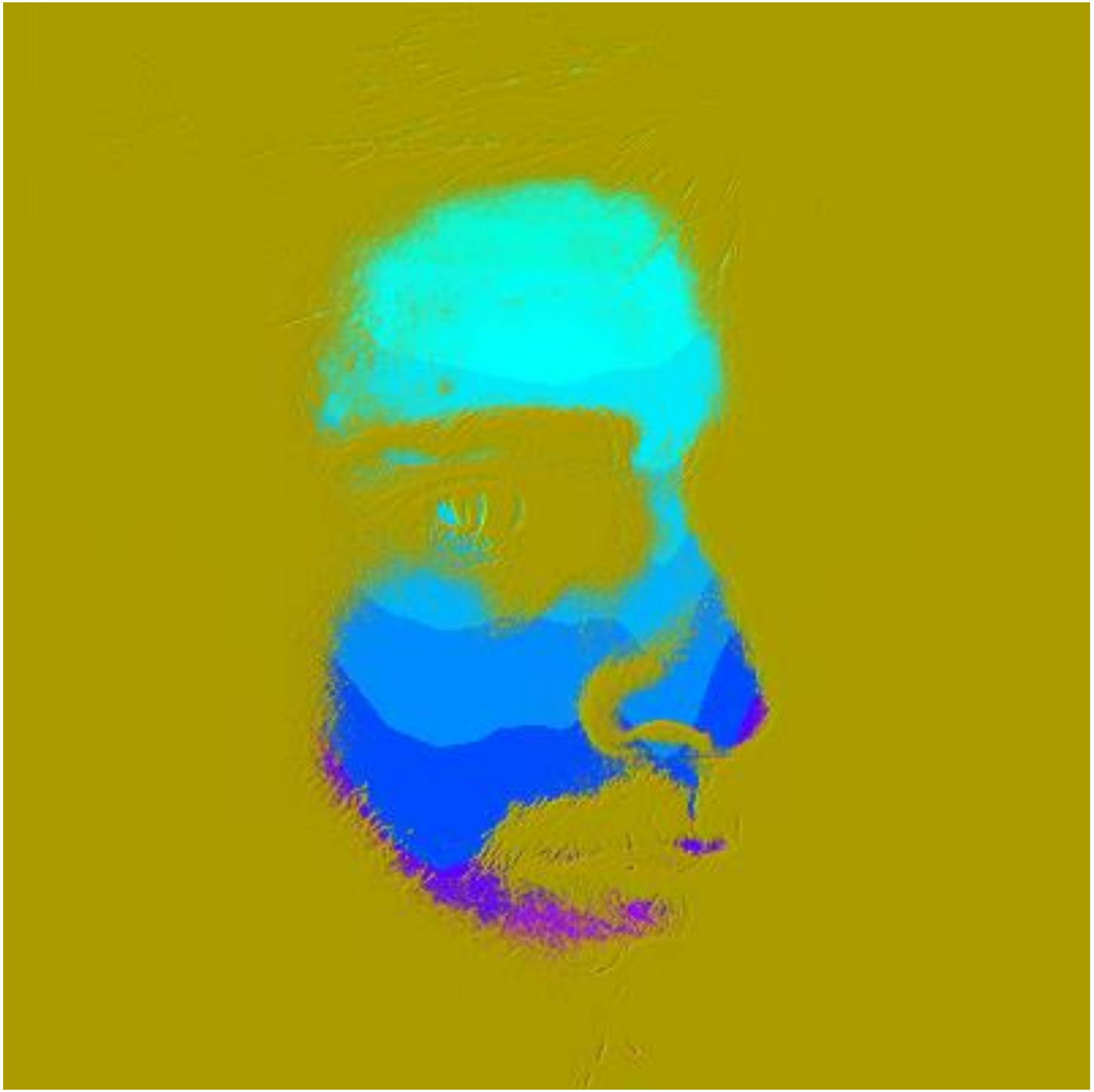


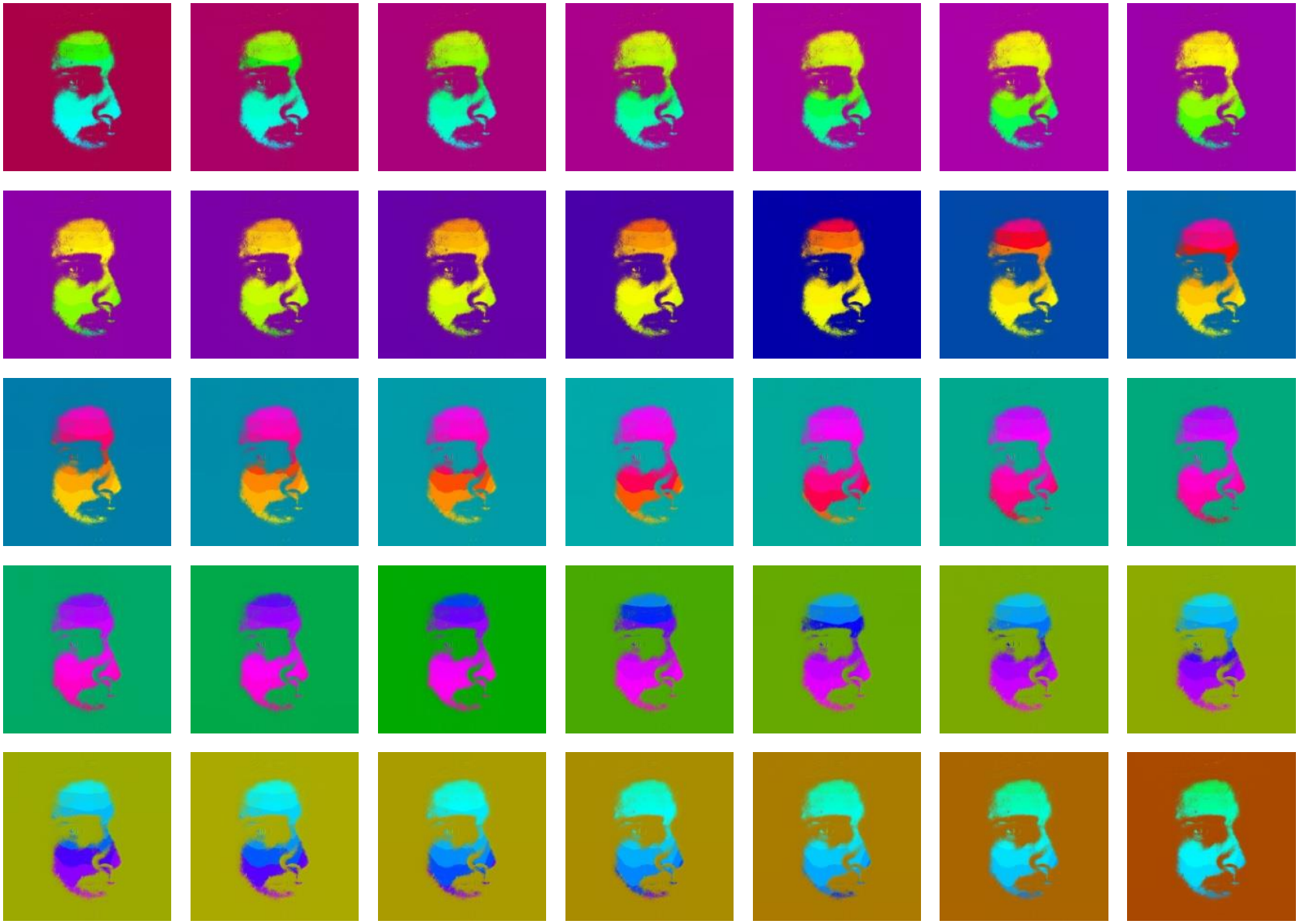
Plus facile en HSV

- Fond :
  - Hue aléatoire
  - Saturation max
  - Luminosité à 0.4
- Visage :
  - Hue + 180, +- 170/nombre de bandes

Génération de couleurs HSV







Problème

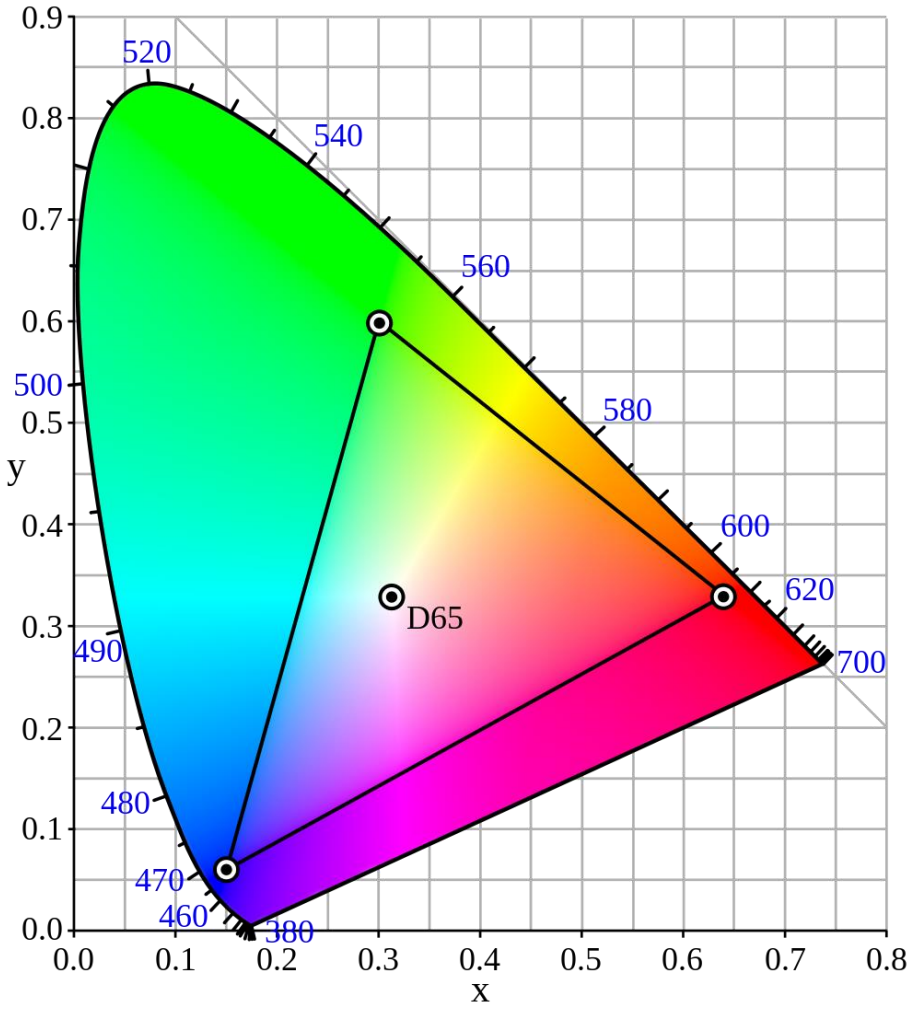
- V ne correspond pas vraiment à la perception de la luminosité ((R+G+B)/3), mais a la quantité de lumière envoyée par le pixel.
- CIE XYZ : Y = Luminosité perçue

$$Y = 0,2126 \cdot r_{lin} + 0,7152 \cdot g_{lin} + 0,0722 \cdot b_{lin}.$$

- Calcule sur un RGB linéaire. Suppose donc correction gamma si lecture dans une texture et avant d'envoyer à l'écran.

CIE XYZ

- x = X/(X+Y+Z)
- y = Y/(X+Y+Z)
- z = 1-x-y;



CIE XYZ

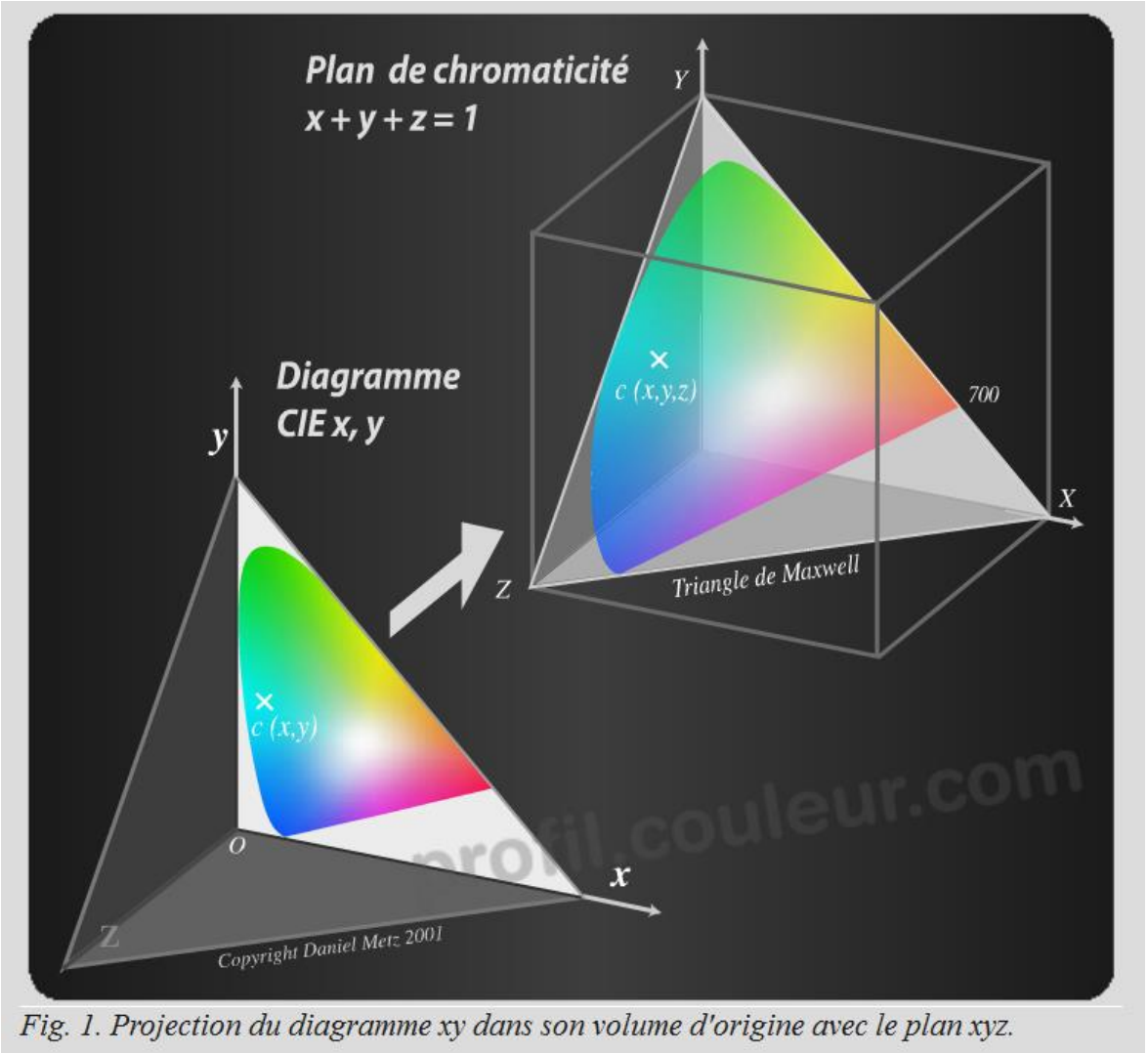
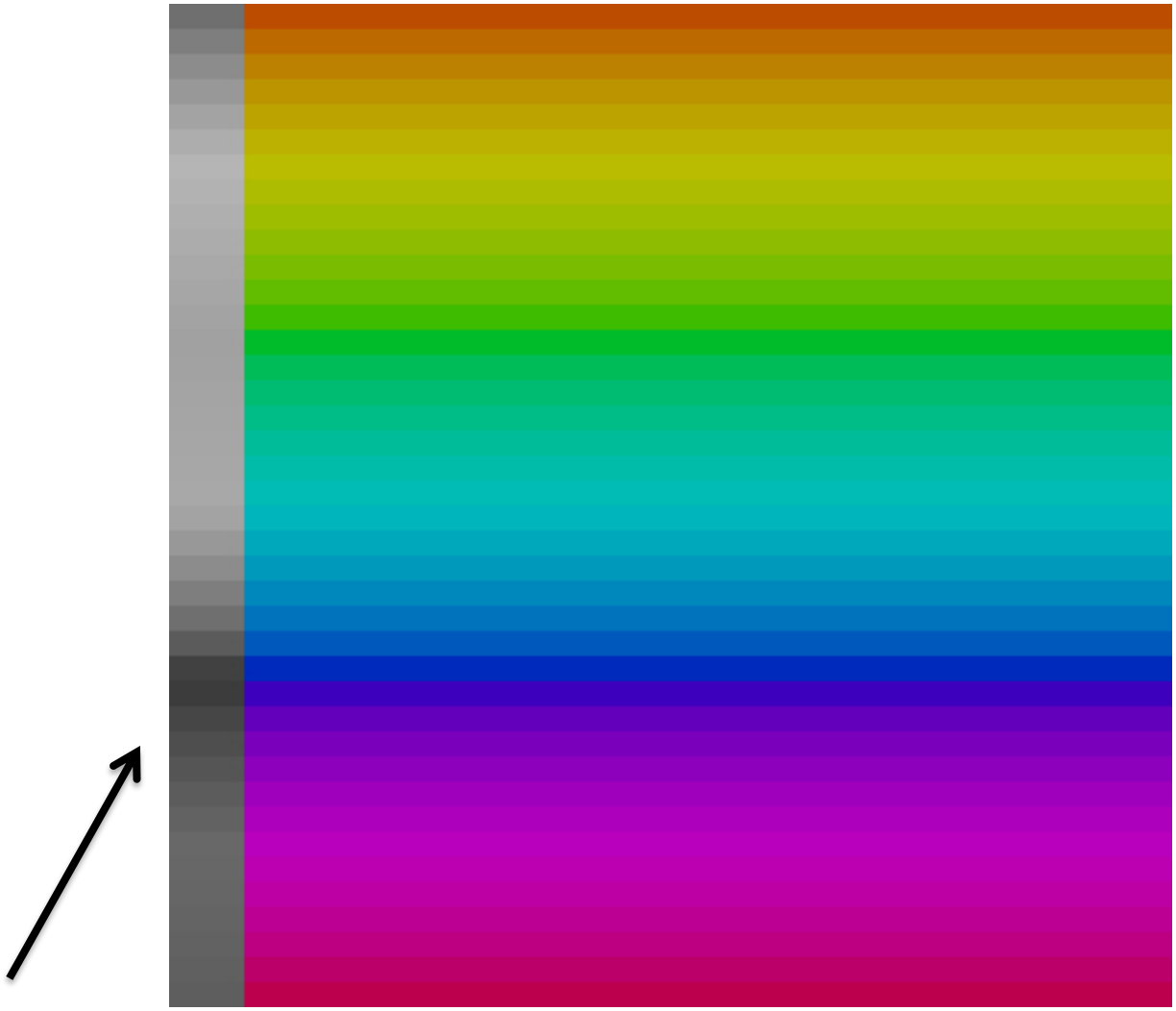


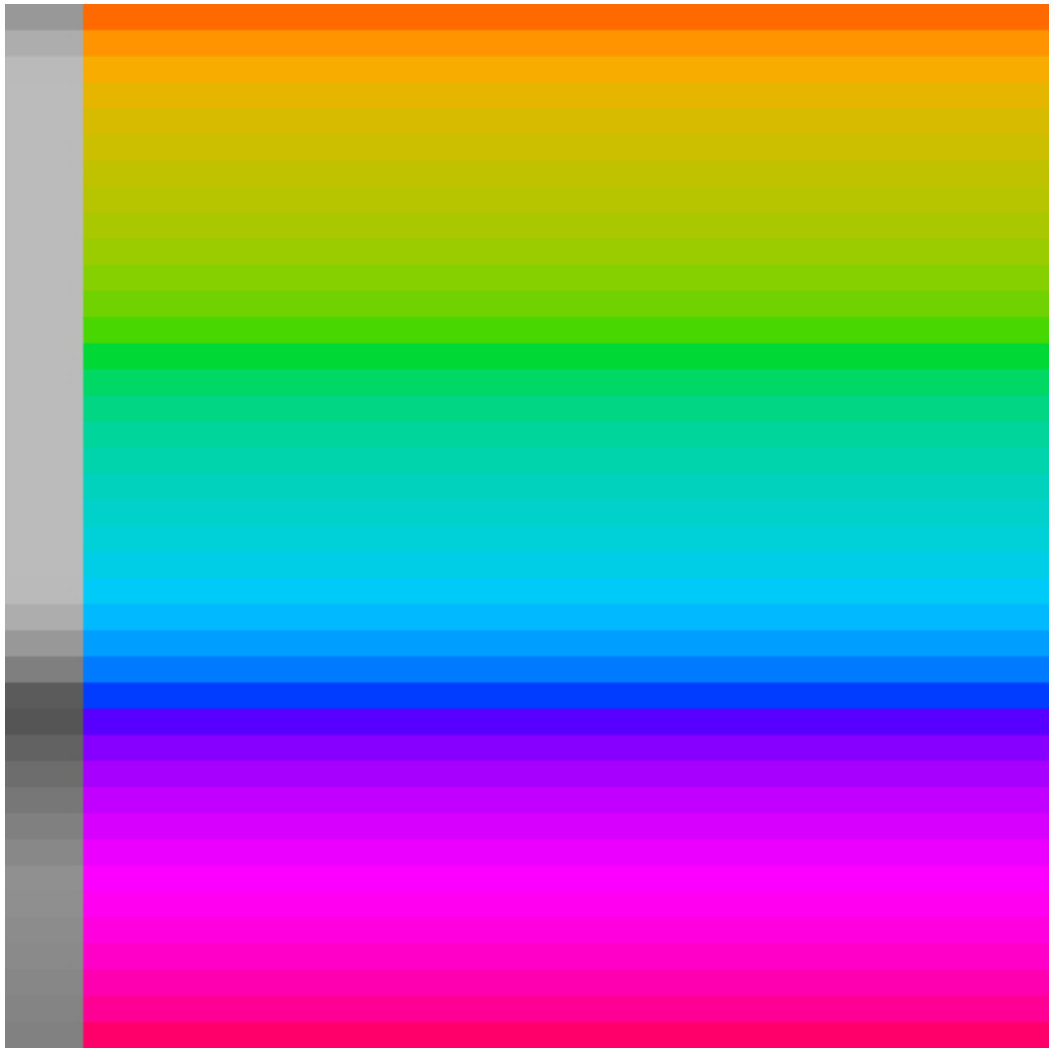
Fig. 1. Projection du diagramme xy dans son volume d'origine avec le plan xyz.

Génération HSV (h:[0:360] s:1 v:0.5)



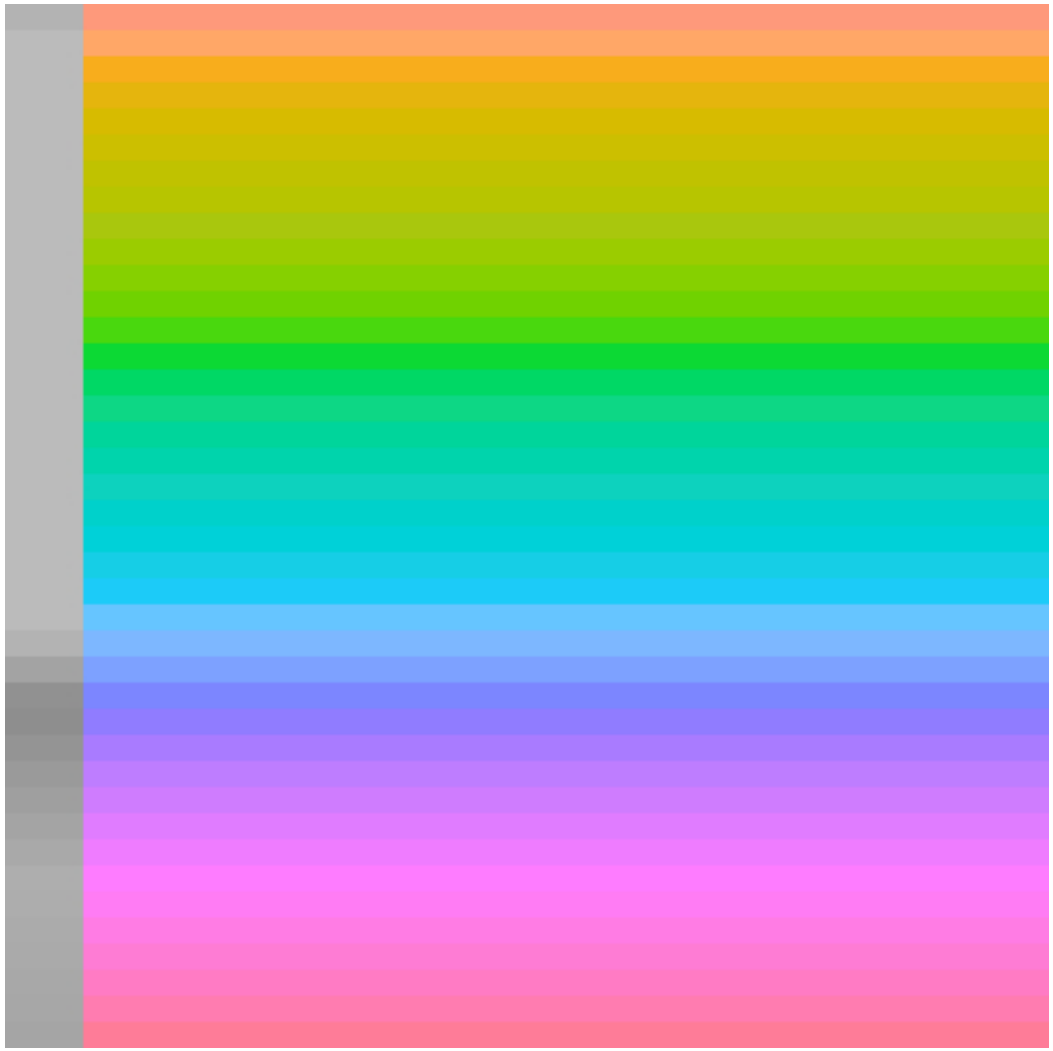
Y * RGB(255,255,255)

Correction v [0,1] pour tendre vers Y = 0.5



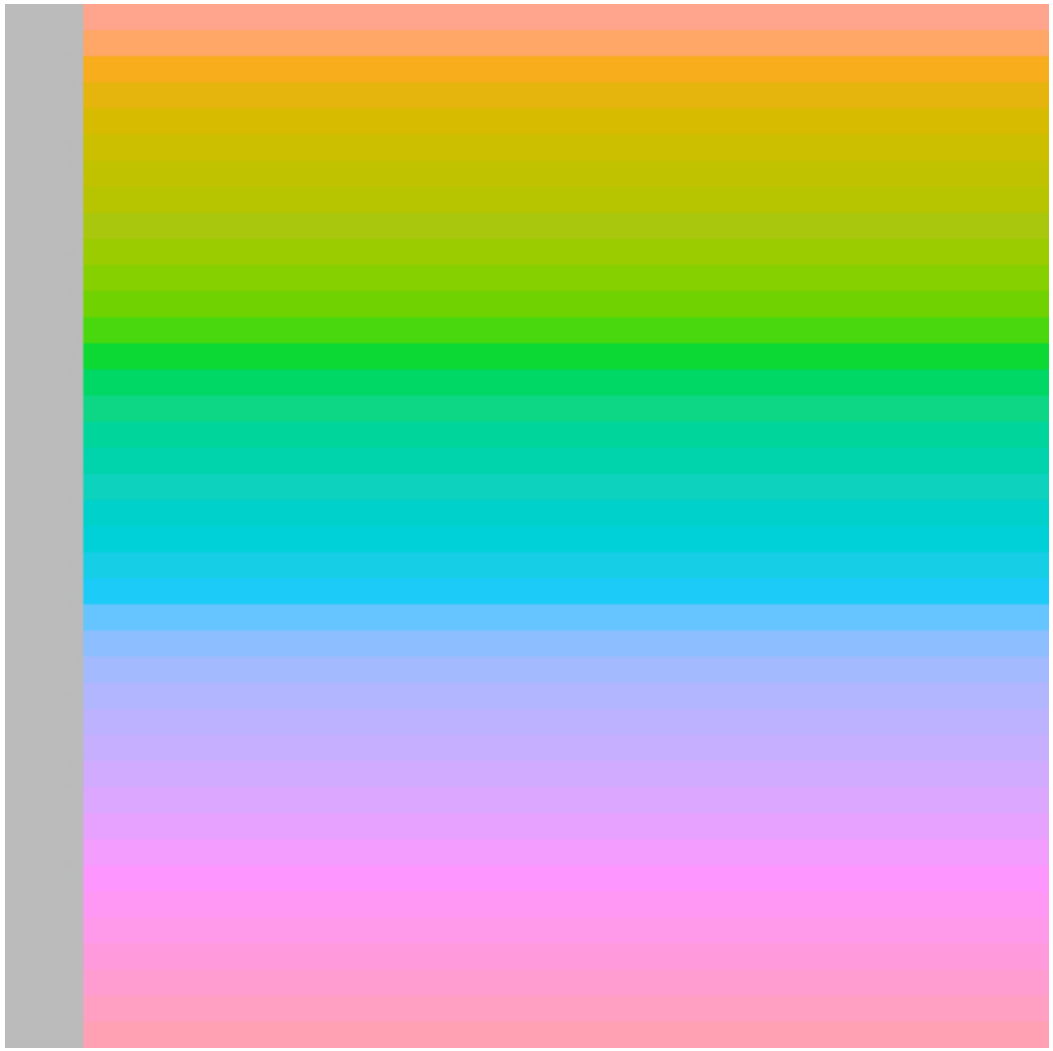
Y * RGB(255,255,255)

Correction v [0,1] et s [s-0.2,s+0.2] pour tendre vers Y = 0.5



Y * RGB(255,255,255)

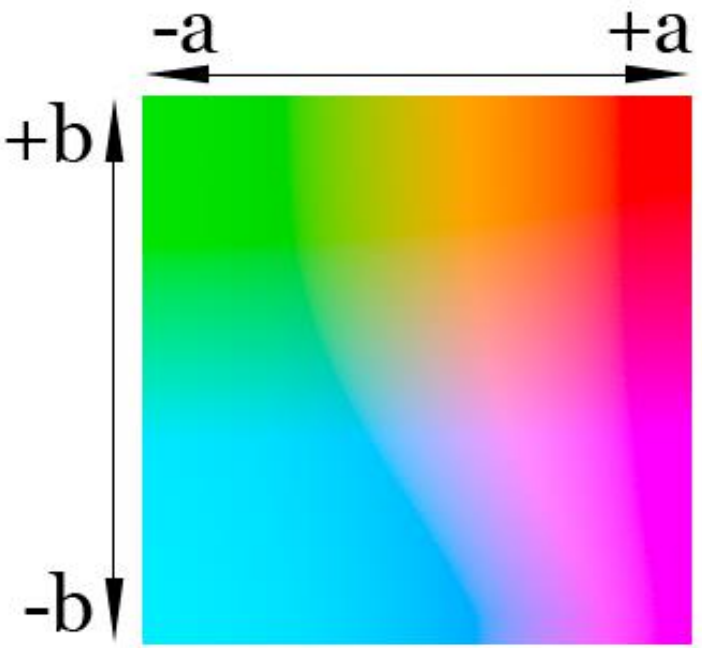
Correction v [0,1] et s [0,1] pour tendre vers Y = 0.5



Y * RGB(255,255,255)

Attention, c’est plus compliqué

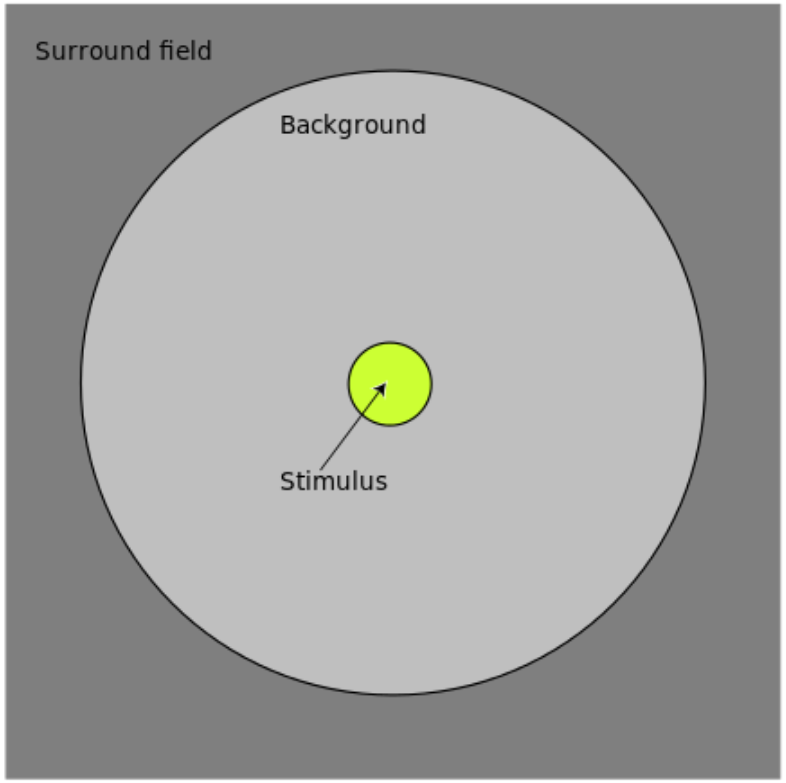
- CIE XYZ ne corrige pas tout les problèmes
- Date de 1931, comme RVB
- Espaces Lab, Luv, U’V’W’
  - Espaces non linéaires, écart entre couleurs plus proches de la vision humaine
  - L correspond toujours à la luminance, calculé à partir de Y



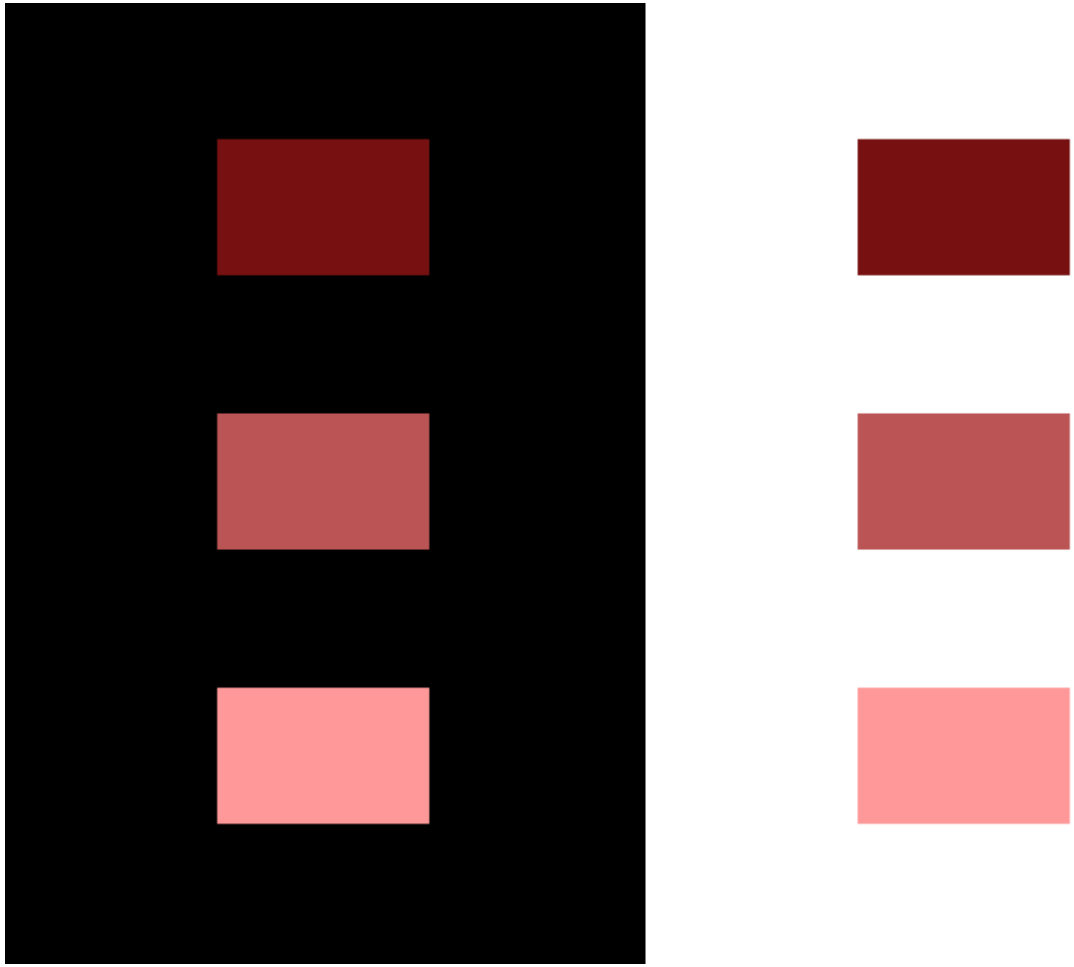
Lab avec L à 75%

Attention, c’est encore plus compliqué

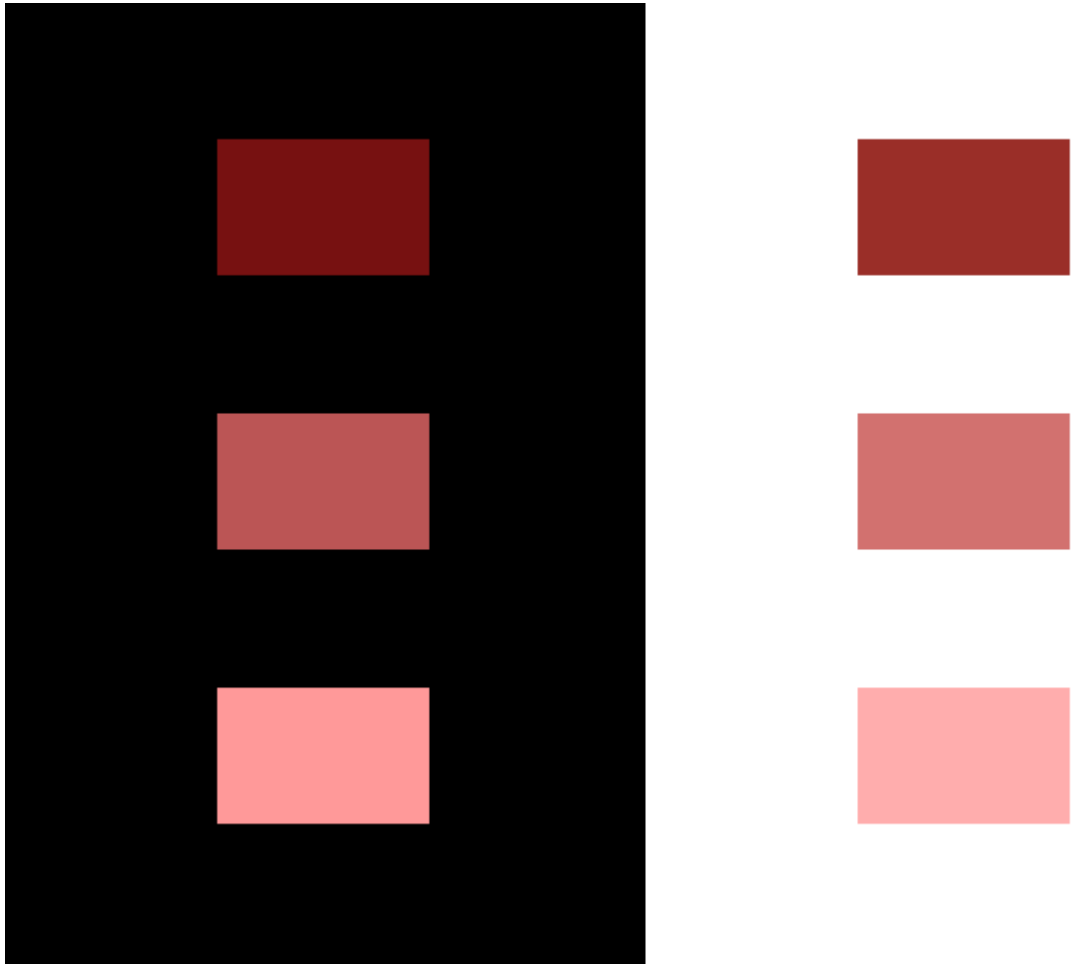
- Lab & co ne corrigent pas tous les problèmes
- Color Appearance Models (ex CIECAM02) 2002



Pixels identiques

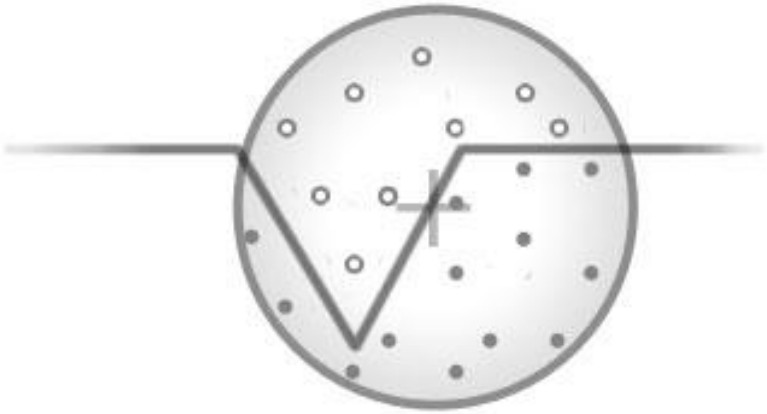


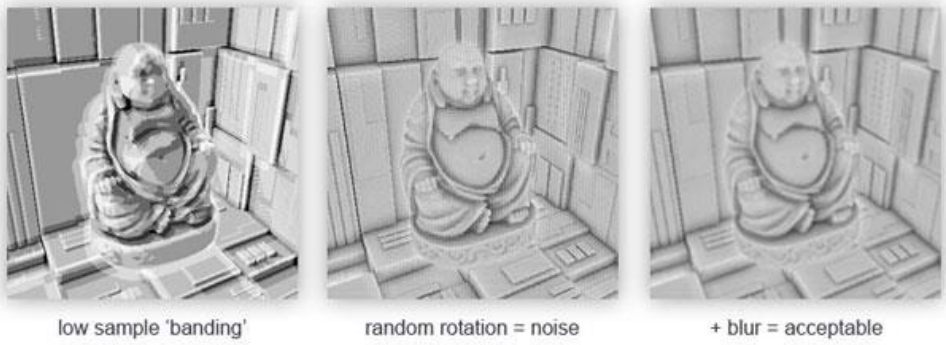
CIECAM02 identiques, pixels différents



SSAO

- Ambient Occlusion : ombre qui provient de l'occlusion de la lumière ambiante, pas d'une source directionnelle
- SSAO : Occlusion ambiante calculée en espace ecran (deffered)
- Crysis method:
  - get fragment view space position (un project avec inverse projection matrix, ou gbuffer de position)
  - project each sample point into screen space to get the coordinates into the depth buffer
  - sample the depth buffer
  - if the sample position is behind the sampled depth (i.e. inside geometry), it contributes to the occlusion factor.
- Voir john-chapman-graphics pour full tutorial.





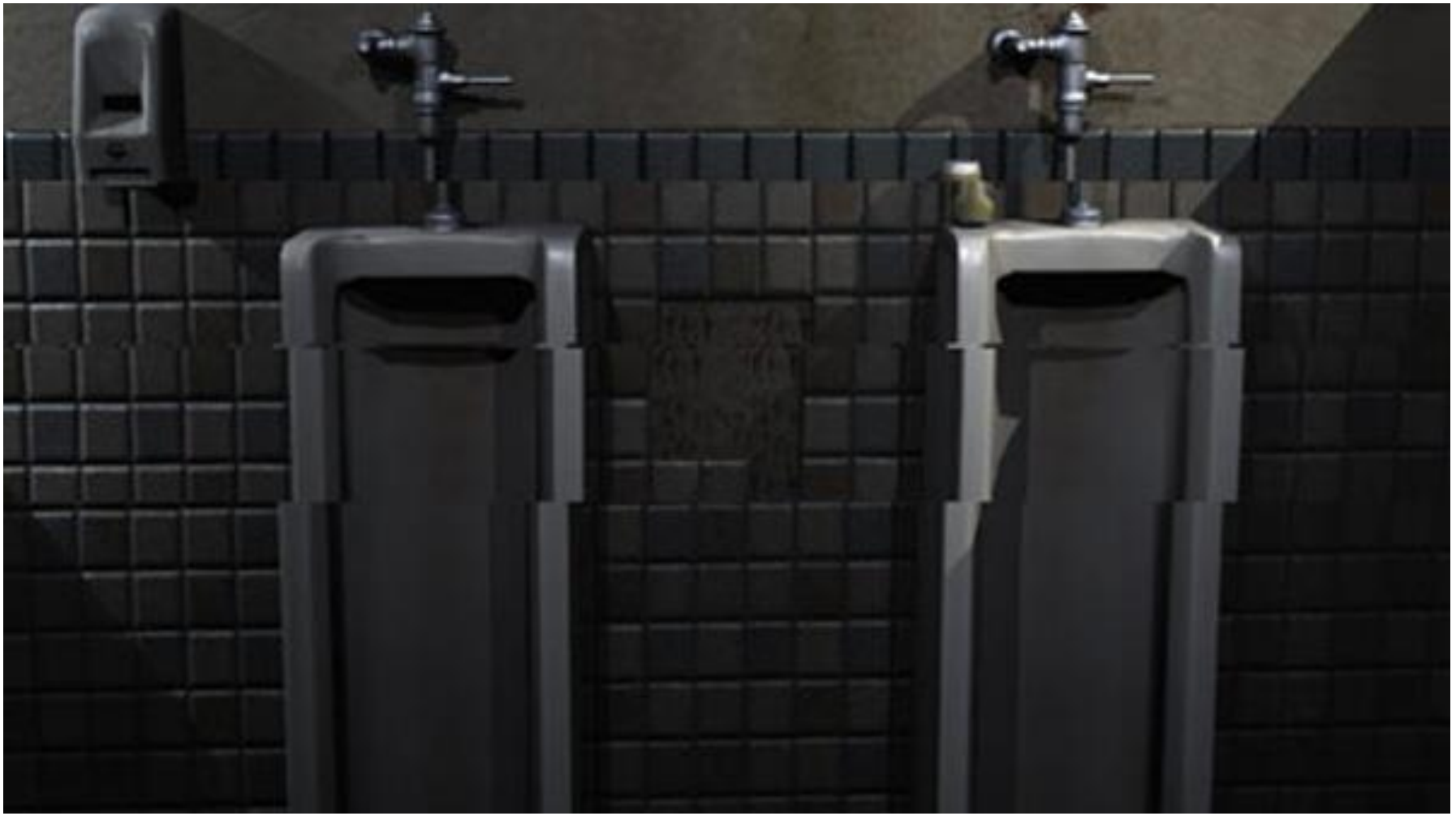
Anti Aliasing

MSAA

FXAA et TXAA

Tearing

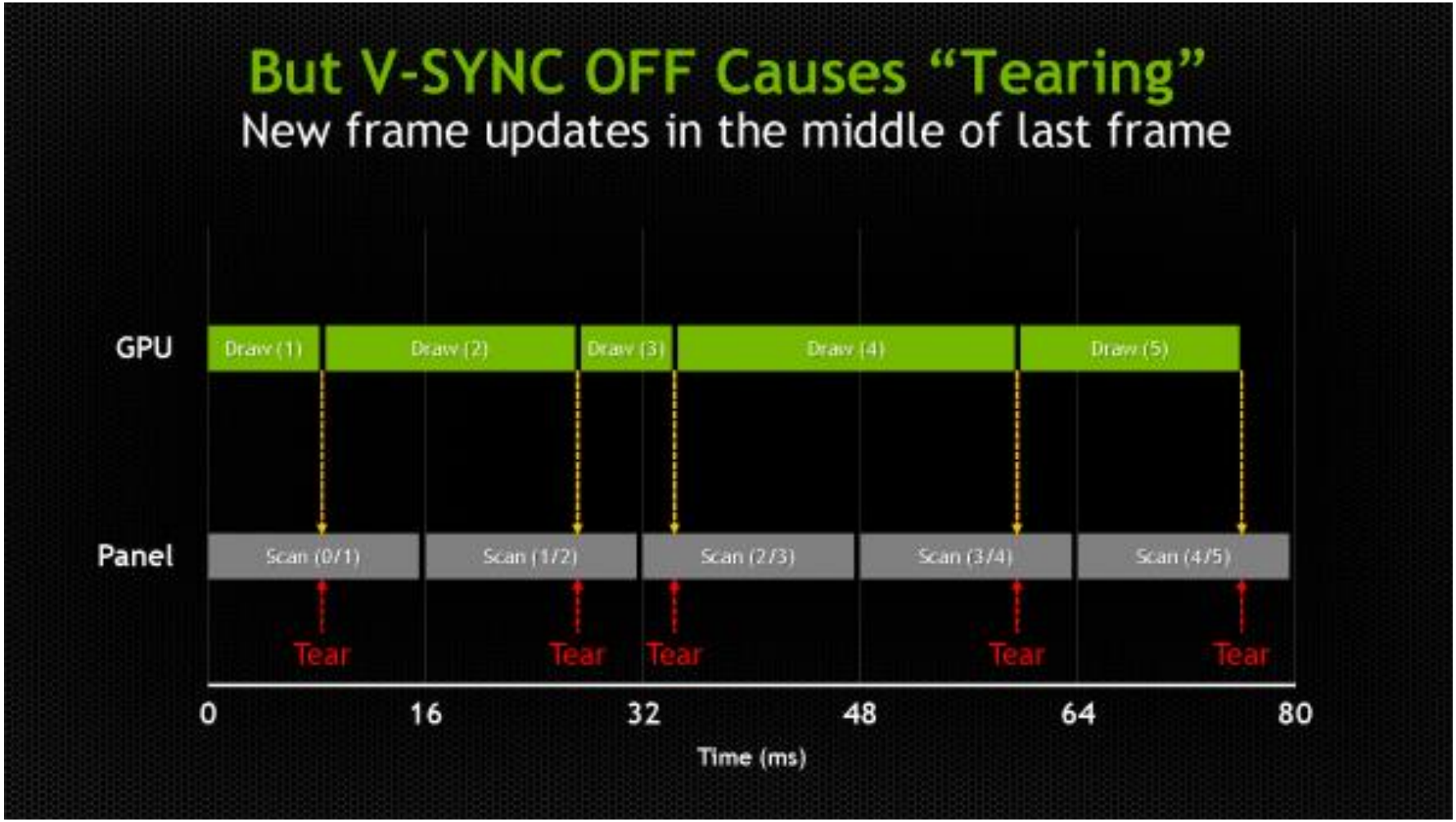
Tearing



Adding Vsync

- Param sur le driver (nvidia)
- Directement depuis le code:
  - Include « wglew.h » pour les extensions gl platerform specifiques
  - wglSwapInterval(1) active la VSync

Tearing


But V-SYNC OFF Causes "Tearing"
New frame updates in the middle of last frame
GPU
Draw (1)
Draw (2)
Draw (3)
Draw (4)
Draw (5)
Panel
Scan (0/1)
Scan (1/2)
Scan (2/3)
Scan (3/4)
Scan (4/5)
Tear
Tear
Tear
Tear
Tear
0
16
32
48
64
80
Time (ms)

Vsync Input Lag