

Visibilité, affichage

Eliminer le parties cachées:

Transformation du modèle

Illumination

Transformation de vue

Découpage (Clipping)

Projection (dans l'espace écran)

Rastérisation

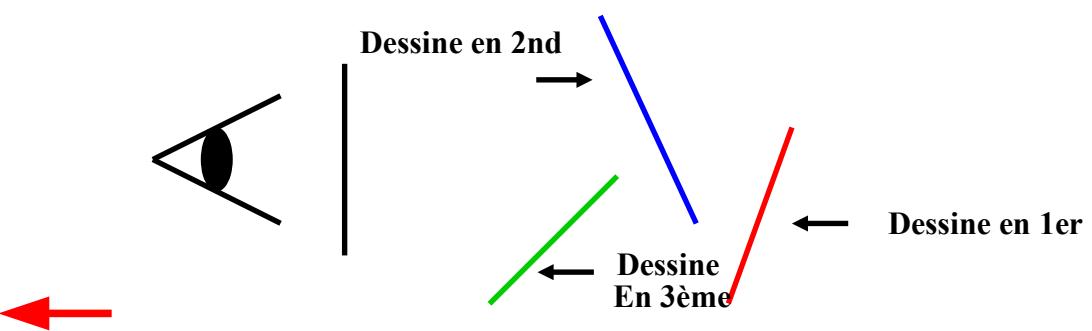
Visibilité/Affichage

• Algorithme du peintre:

- Tri des objets selon la profondeur (Z)
- Boucle sur les objets depuis l'arrière plan jusqu'au premier plan

• $n \log(n)$

• pb dans certain cas



Visibilité, affichage

Transformation du modèle

Illumination

Transformation de vue

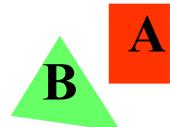
Découpage (Clipping)

Projection (dans l'espace écran)

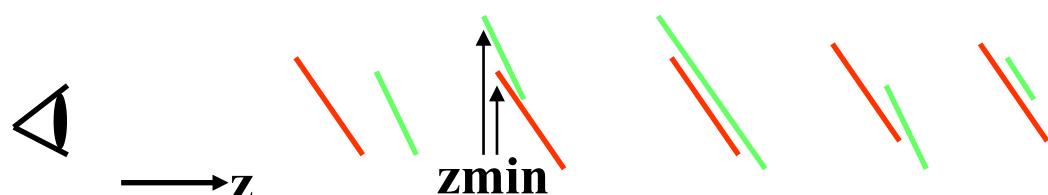
Rastérisation

Visibilité/Affichage

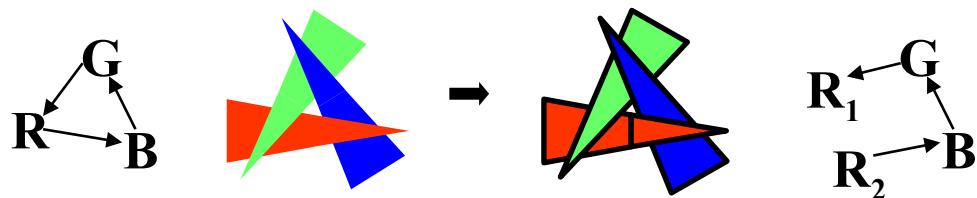
- L'ordre de profondeur est un *ordre partiel*. Les configurations sont :



- Trier les objets en fonction du zmin ne fonctionne pas toujours ! (idem pour zmax)



- Quand l'ordonnancement est cyclique : découper les objets!

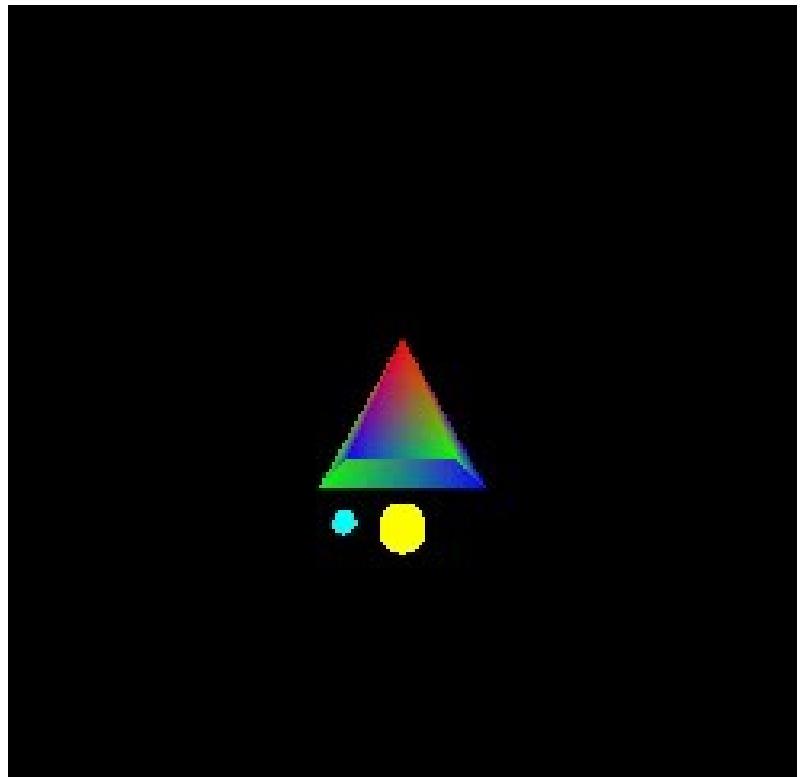


En OpenGL

```
while (1) {  
    glClear(GL_COLOR_BUFFER_BIT);  
    draw_3d_object_A();  
    draw_3d_object_B();  
}
```

B est toujours dessiné
en second!

Donc par dessus A...



Algorithme du Z-Buffer

- Initialisation

Boucle sur x,y

zbuf[x,y] = infini

- Etapes du rendu

Boucle sur tous les objet {

Projection d'un objet (boucle sur x,y) {

// calcul du z de l'objet courant par rapport au pixel (x,y) et test de la profondeur

si $z(x,y) < zbuf[x,y]$

// mise à jour du z-buffer

zbuf[x,y] = z(x,y)

// mise à jour de l'image (typiquement RGB)

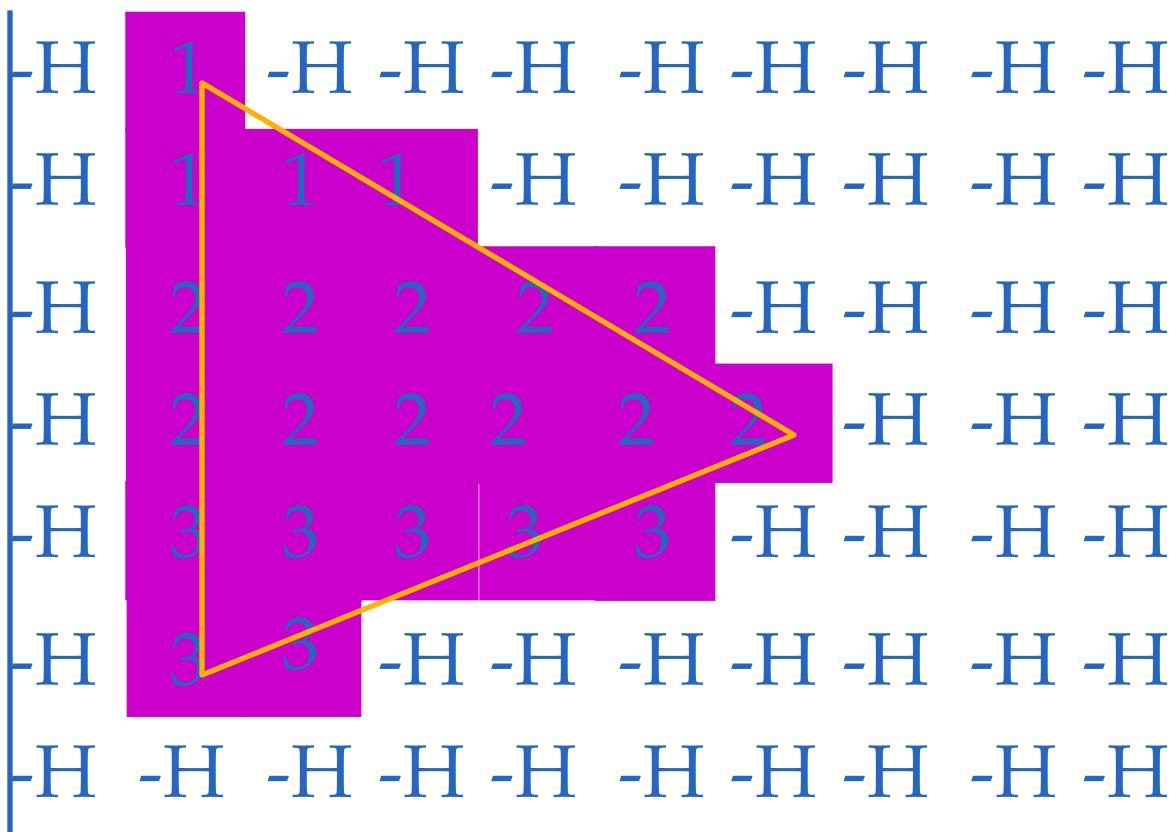
image[x,y] = illumination(x,y)

}

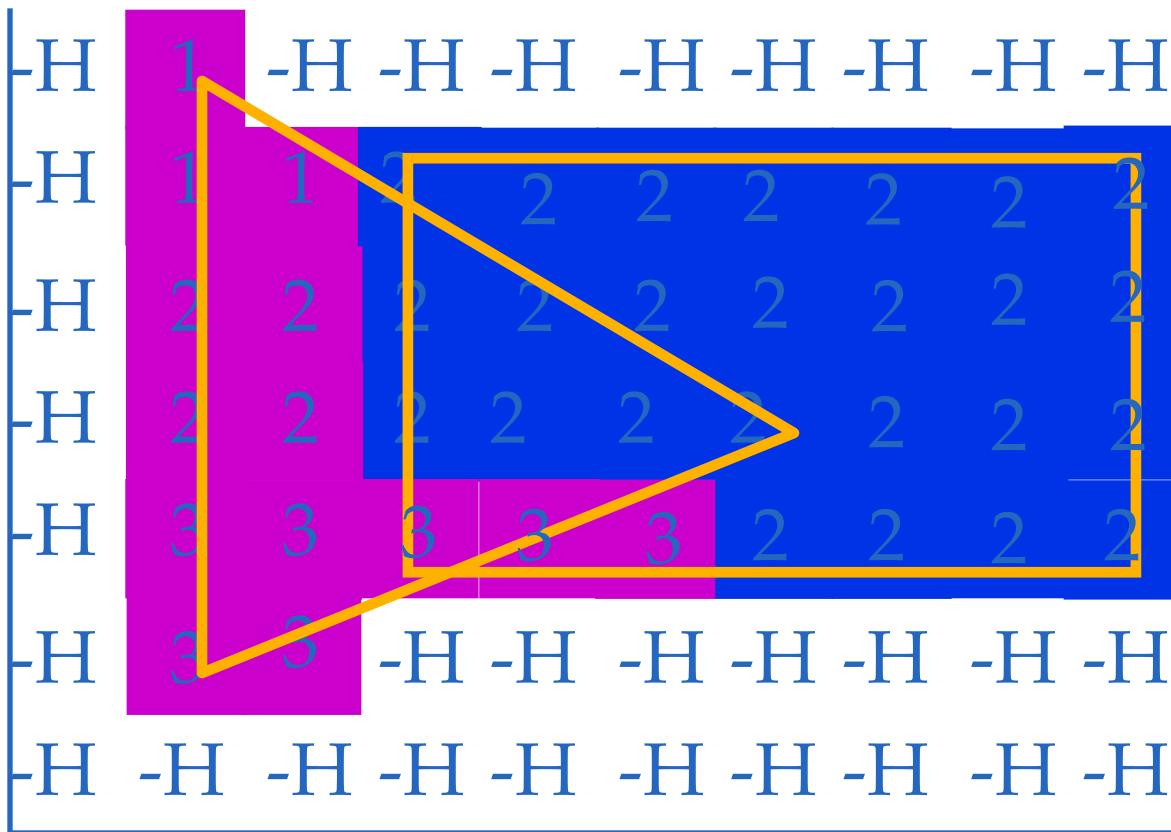
Algorithme du Z-Buffer

-H
-H
-H
-H
-H
-H
-H
-H

Algorithme du Z-Buffer



Algorithme du Z-Buffer



Z-Buffer en OpenGL

- Initialisation

```
glutInitDisplayMode ( GLUT_DOUBLE | GLUT_RGB  
| GLUT_DEPTH );  
  
glEnable(GL_DEPTH_TEST);
```

- Etapes du rendu

```
glClear (GL_COLOR_BUFFER_BIT |  
GL_DEPTH_BUFFER_BIT)
```

