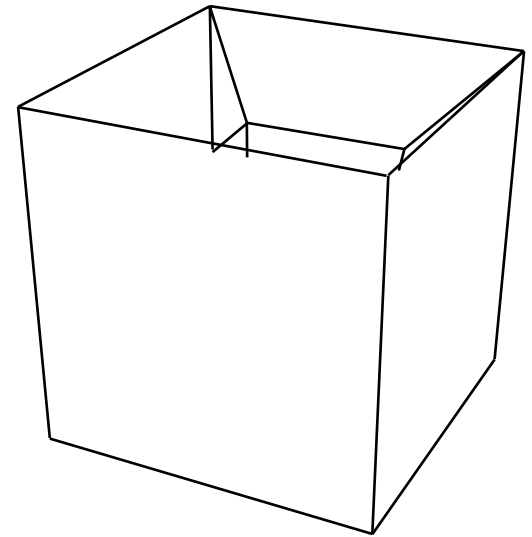
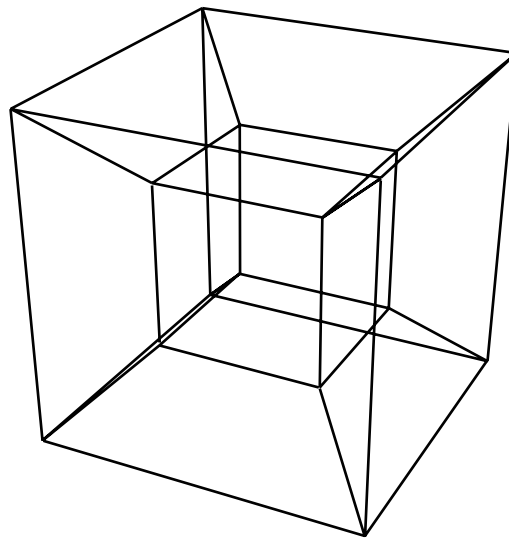
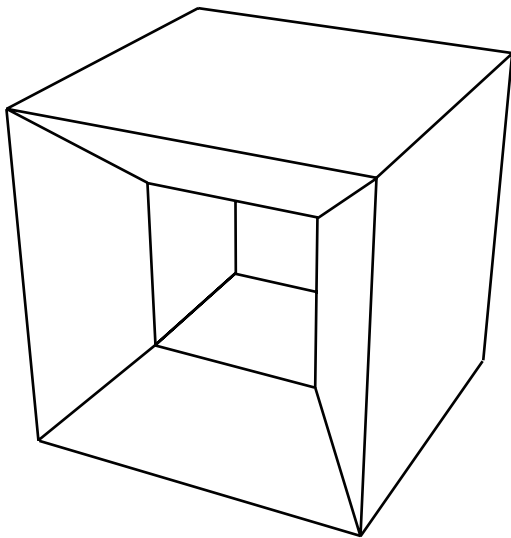


ELIMINATION des PARTIES CACHEES

1. Généralités

👉 nécessité



👉 choix d'un algorithme :

👉 nature des données :

surfaces planes ou gauches ?

objets définis par arbre de construction ?

objets intersectants ?

...

👉 résultat :

tableau de pixels ?

contours ?

image de haute qualité ?

temps réel ou pas ?

coût ?

🔑 fondamentalement : tri en 3 dimensions

👉 tri de points









👉 tri de segments

👉 tri de polygones

	objet	image
dimension 0	tracé de rayons	Z-buffer
dimension 1	Watkins	
dimension 2		Warnock
dimension 3	Fuchs	Newell



utilisation de la cohérence :

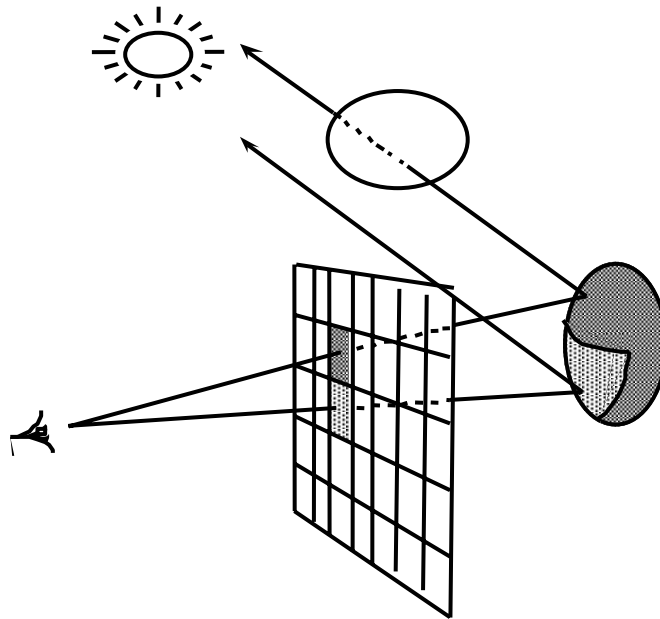
-  d'image
-  d'objet
-  de face
-  d'arête
-  de balayage de ligne
-  d'aire
-  de profondeur
-  ...

2. Tri en dimension 0

2.1- Tracé de rayons

principe : simulation des lois de l'optique

modèle de l'appareil photographique inversé



Algorithme :

```
fonction calcul_couleur(rayon) → couleur
{ couleur_locale ← couleur_ambiante;
  pour chaque objet faire
    { calculer_intersection(rayon, objet);
      garder-objet_le_plus_proche;
    }
  si un objet a été gardé alors calculer_couleur_locale;
  si réflexion alors couleur_r ← calcul_couleur(rayon_réfléchi);
  si transmission alors couleur_t ← calcul_couleur(rayon_transmi);
  retourner(couleur_locale+couleur_r+couleur_t);
}

programme principal
{ pour chaque pixel (i,j) de l'écran faire
  { rayon ← calculer_vecteur(œil,pixel);
    afficher(calcul_couleur(rayon));
  }
}
```

=> évaluation d 'un arbre des rayons

☛ avantages :

- ☞ basé sur un modèle optique de la lumière, il permet facilement de simuler réflexions, réfractions, ombres portées ...
- ☞ accepte les arbres de construction
- ☞ simple : le point central est constitué par les intersections droite-primitive

☛ inconvénients :

- ☞ temps de calcul

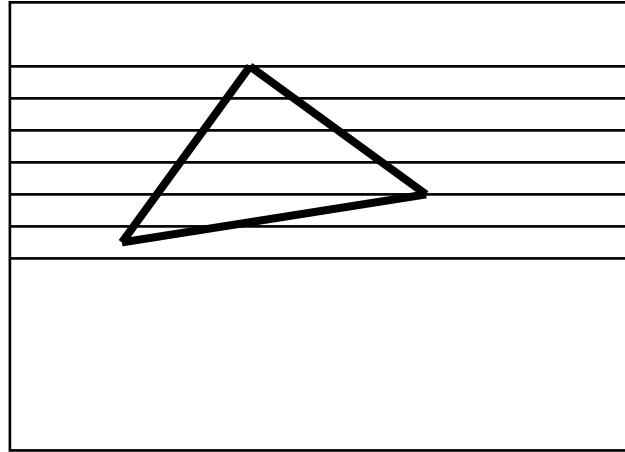
2.2- Le Tampon de Profondeur (z-buffer)

principe : tri suivant la direction des z et utilisation d'une mémoire tampon, en plus de la mémoire d'image

algorithme :

```
{ pour chaque pixel (i, j) de l'écran faire
  { Z[i, j] ← infini ;
    coul[i, j] ← coul_fond ;
  }
  { pour chaque pixel (i,j) en lequel E se projette faire (*)
    { z ← profondeur de E en (i,j) ;
      si z < Z[i,j] alors
        { Z[i,j] ← z ;
          coul[i,j] ← couleur de E en (i,j) ;
        }
      }
    }
  }
}
```


(*) si E est un polygone, ceci correspond à un remplissage :



☛ avantages :

☞ simplicité => câblage

☞ inutilité de tout tri

☛ inconvénients :

☞ transparence

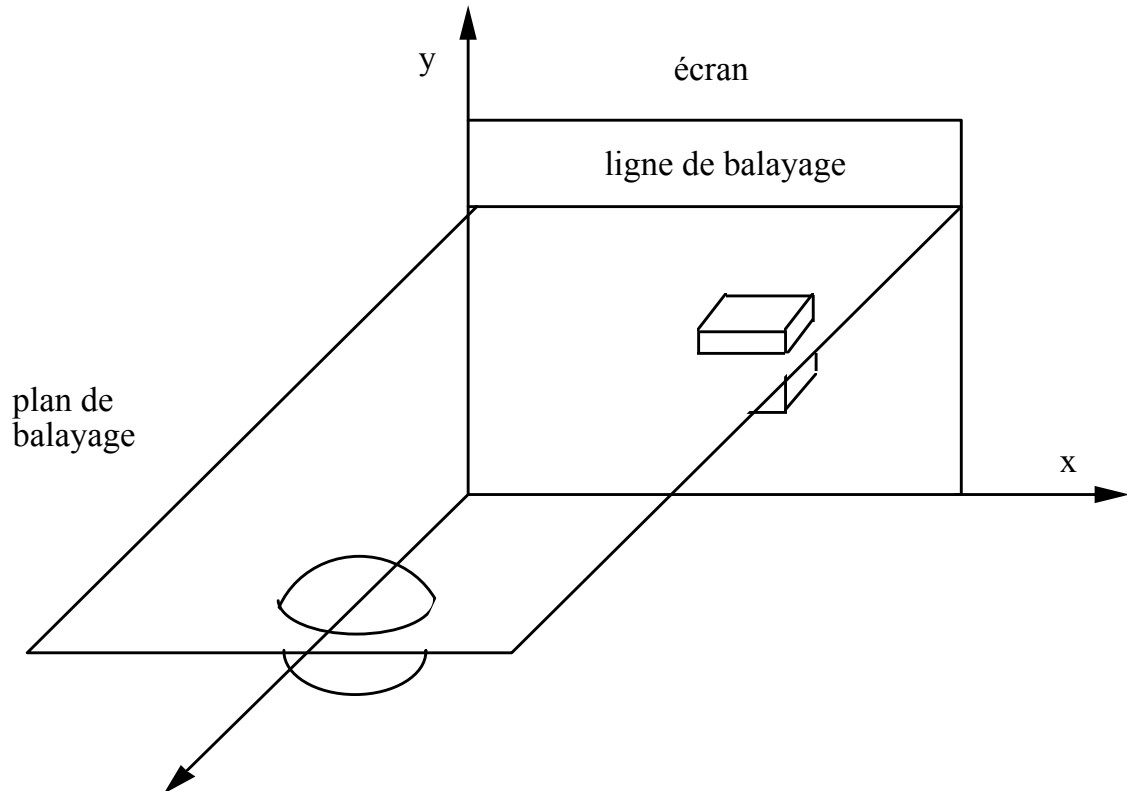
☞ antialiassage

☞ modélisation par arbre de construction

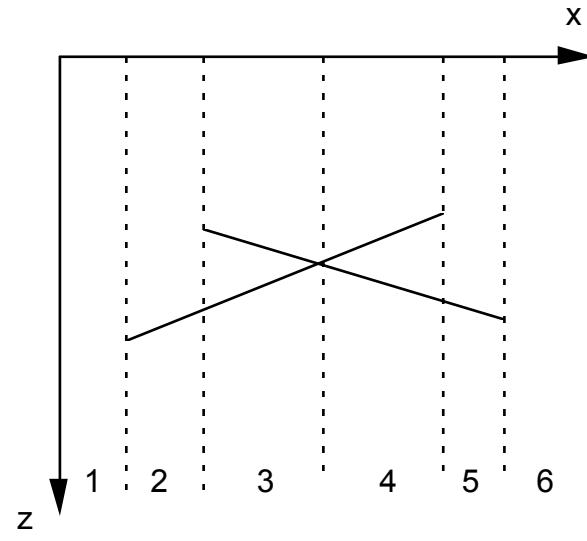
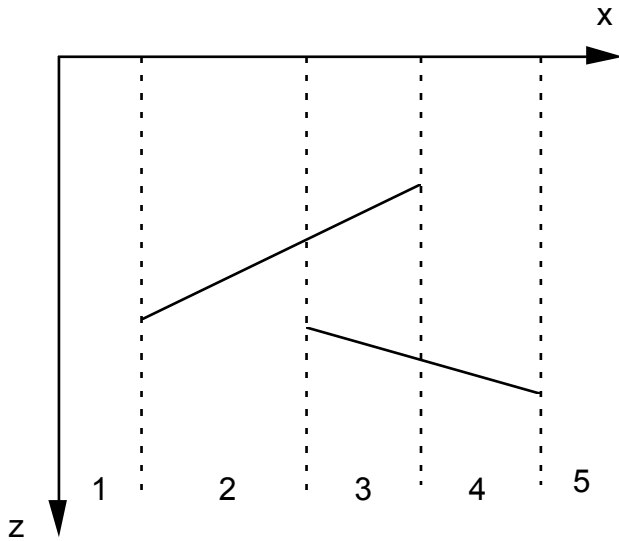
☞ (place mémoire)

3. Tri en dimension 1 : algorithmes de balayage ligne par ligne (scan-line)

- principe : décomposition de la scène à visualiser par des plans perpendiculaires à l'écran passant par les lignes de balayage



🔑 méthode des empans (Watkins 1970)



algorithme :

{ **pour** chaque ligne de balayage **faire**

{ calculer l'intersection de la scène avec le plan de balayage ;

ordonner les segments trouvés suivant les x ;

calculer les empan successifs et créer une liste L d'empan;

pour chaque empan e de L **faire**

{ L ← L-e ;

déterminer le segment visible à gauche et le segment visible à droite ;

si ce sont les mêmes **alors** afficher segment ;

sinon

{ calculer l'intersection entre les segments ;

créer un empan gauche et un empan droite et les insérer dans L ;

}

}

}

}

utilisation de la cohérence de ligne et de la cohérence d'arête

☛ extension d'Atherton :

s'applique aux scènes modélisées à l'aide d'un arbre de construction ;

la sélection du segment visible se fait en évaluant l'arbre de construction par une procédure récursive.

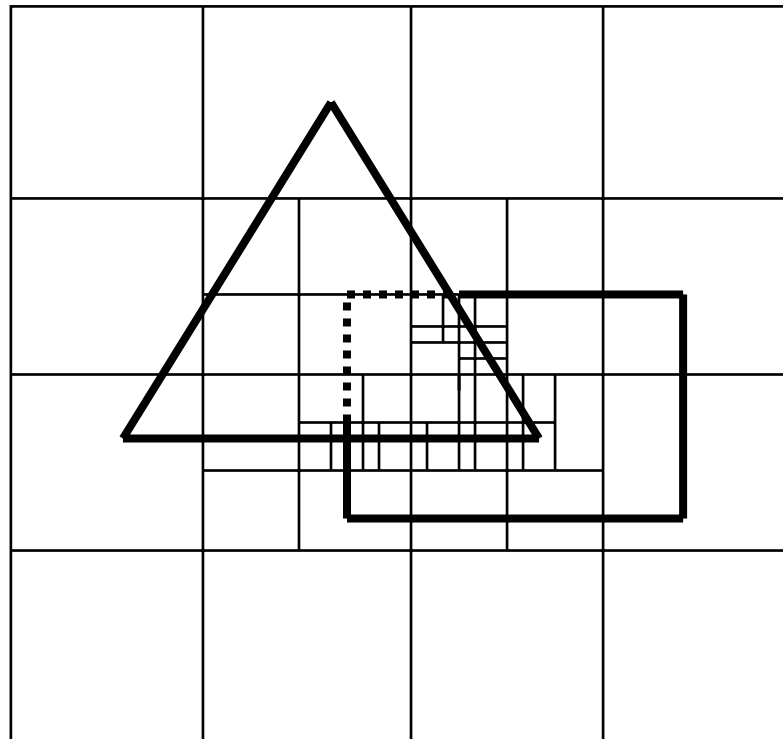
☛ extension aux surfaces :

- facettisation
- Blinn
- Whitted
- Lane - Carpenter

4. Tri en dimension 2 : Algorithmes de subdivision

principe : subdivision de l'écran en quatre parties, jusqu'à ce que l'on aboutisse à une situation simple;

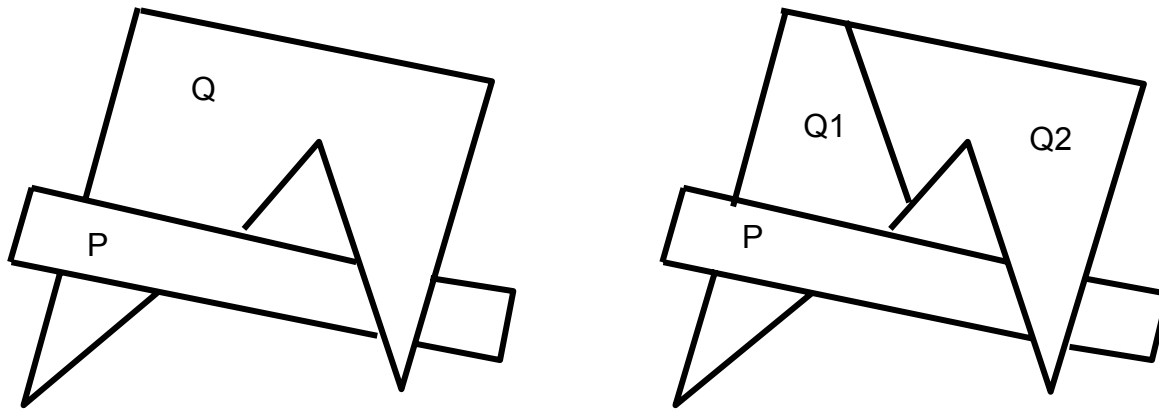
Warnock (1969) : subdivision de l'écran



5. Tri en dimension 3 : algorithmes avec listes de priorité

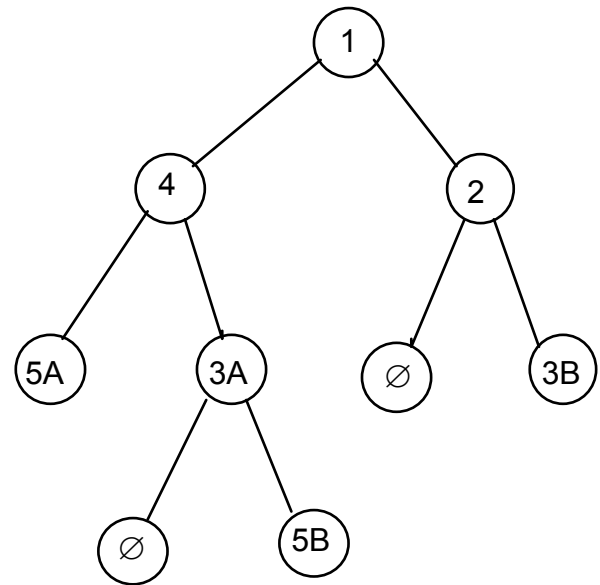
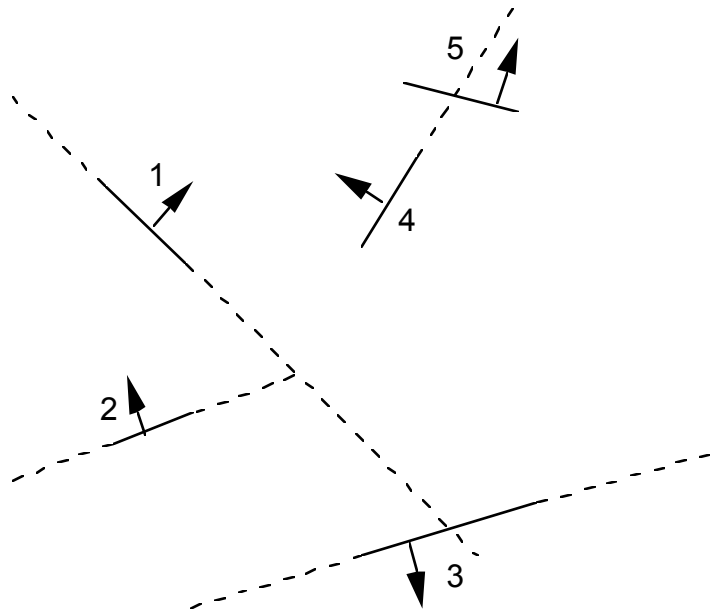
principe : précalcul, dans l'espace objet, d'un ordre de visibilité (ou de priorité);

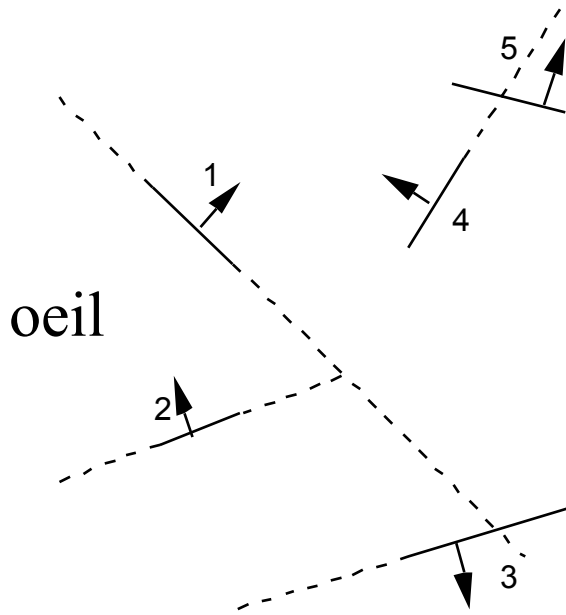
Newell, Newell, Sancha (1972)



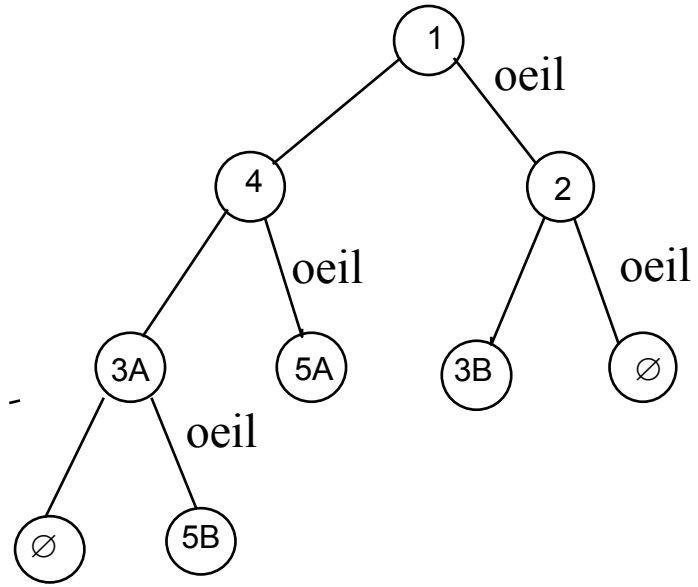
Shumacker (1969)

arbre de partition spatiale : Fuchs (1980)





oeil



3A 5B 4 5A 1 3B 2