

# Chapitre 7 : Lancer de rayons

## Modélisation 3D et Synthèse

Fabrice Aubert  
fabrice.aubert@lifl.fr



IEEA - Master Info - Parcours IVI

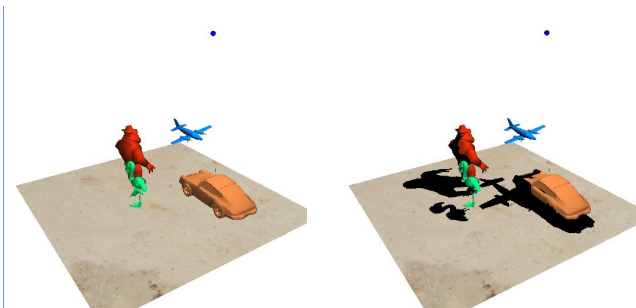
2012-2013

# 1 Introduction : modèles d'illuminations

- ▶ Modèle local : on calcule l'intensité en un point donné d'une surface en ne tenant compte que de l'intensité des sources lumineuses. Exemple : modèle de Phong (introduit au chapitre précédent).
- ▶ Modèle global : on tient compte de la contribution de toutes les surfaces pour l'intensité incidente (échanges lumineux entre toutes les surfaces). Exemple : lancer de rayons (pseudo-global), radiosit .

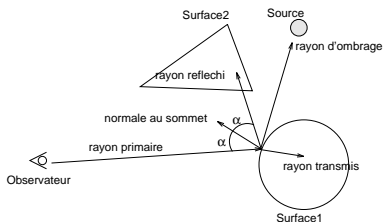
# Remarques sur modèles locaux : ombres portées

- ▶ Il s'agit, au point considéré, de l'occultation d'une source lumineuse par d'autres éléments de la scène.
- ▶ Objet récepteur : l'objet qui subit l'ombre d'un autre objet (celui qui "reçoit" l'ombre portée).
- ▶ Objet émetteur : l'objet qui provoque l'ombre.
- ▶ Différentes méthodes : shadow map, depth map, shadow volumes, ...

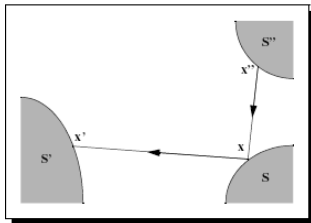


# Principe du lancer de rayons

- ▶ Suivre le chemin inverse de la lumière (but : calculer la couleur en chacun des pixels de l'écran).
- ▶ Dans sa version simple, on détermine pour chaque point rencontré un éclairage local (i.e. éclairage direct par les sources) auquel on ajoute l'éclairage provenant d'un rayon réfléchi (objet miroir) et d'un rayon transmis (réfraction)  $\Rightarrow$  Modèle de Whitted (80).



# Modèle global : équation de Kajiya (86)



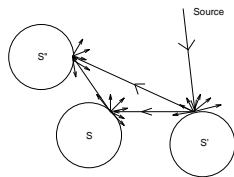
$$I(x, x') = g(x, x') \left[ \epsilon(x, x') + \int_A \rho(x, x', x'') I(x'', x) dx'' \right] \quad (1)$$

avec

- ▶  $I(x, x')$  l'intensité véhiculée de  $x$  vers  $x'$ ,
- ▶  $g(x, x')$  la fonction de visibilité entre les points  $x$  et  $x'$  (0 s'il ne se voient pas, sinon  $g$  varie comme l'inverse du carré de la distance entre  $x$  et  $x'$ ),
- ▶  $\epsilon(x, x')$  l'intensité propre transférée de  $x$  vers  $x'$ ,
- ▶  $\rho(x, x', x'')$  est la réflectance bi-directionnelle au point  $x$  correspondant aux directions  $x'$  et  $x''$ .
- ▶  $A$  ensemble des surfaces constituant la scène (remarque :  $S'$  fait partie de  $A$ ).

# Méthode de Radiosité

- Résoudre l'équation de Radiance en considérant des reflectances parfaitement diffuses uniquement.



step 1



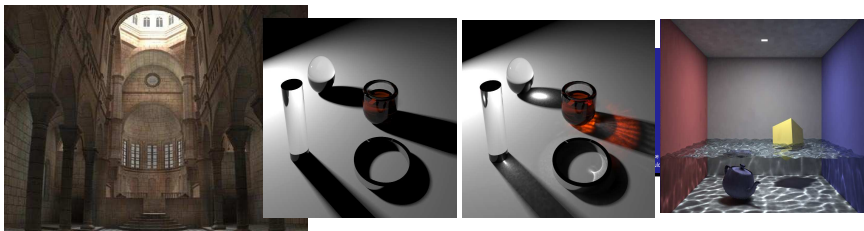
step 79



Source : <http://dudka.cz/rrv>

# Autres méthodes

- ▶ Méthodes hybrides lancer de rayon (spéculaire, transparence) + radiosité (diffus)
- ▶ Radiosité étendue (résolution de l'équation de kajiya avec modèle BRDF).
- ▶ Photon mapping (diffuser des « photons » en partant des sources ; stockage directement sur les surfaces de l'énergie reçue ; rediffusion pour le spéculaire ; puis lancer de rayons en calculant l'éclairage selon l'énergie stockée (« autour du point d'intersection » ).
- ▶ Modèle volumétrique pour simulation d'atmosphère.

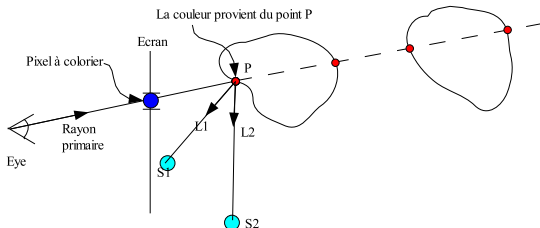




## 2 Mise en oeuvre d'un lancer de rayons

- ▶ Considérer que la « lumière » est portée par des droites (rayon lumineux).
- ▶ Chaque pixel de l'image résultante doit être affecté avec une couleur.
- ▶ Cette couleur de pixel provient de l'éclairement de « quelque chose ».
- ▶ Cet éclairement est porté par une droite qui atteint l'œil (et passe par le pixel dont on cherche la couleur).
- ▶  $\Rightarrow$  Lancer de rayon :
  - Suivre le chemin inverse de la lumière en partant de l'œil.
  - Rayon = droite définie par (œil, pixel) (rayon dit primaire).
  - Chercher la provenance en trouvant l'intersection de ce rayon avec la scène.

# Rayon primaire

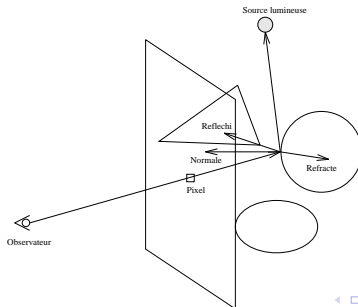


- ▶ Une fois le point  $P$  trouvé (i.e. intersection la plus proche de l'oeil et devant l'écran), on calcule sa couleur.
- ▶ On peut se contenter d'un éclairage local (calcul par Phong par exemple avec les directions d'éclairage  $L1$  et  $L2$ ).
- ▶ Remarque : il faut retenir l'objet dont  $P$  appartient pour avoir la normale, les caractéristiques de matériaux, etc...
- ▶ Remarque : le principe contient intrinsèquement l'élimination des parties cachées (la couleur provient de l'intersection la plus proche).
- ▶ Problème principal du lancer de rayon : calculer l'intersection d'une droite avec la scène et prendre la plus proche.

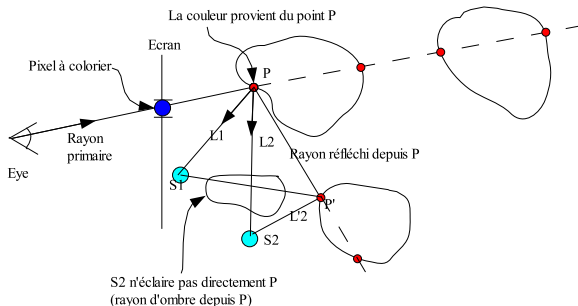
- ▶ Une fois le principe des rayons (et l'intersection avec la scène) mis en place, on peut aisément compléxifier l'éclairage. L'éclairage de  $P$  provient (éventuellement) de :
  - L'éclairage direct par une source (éclairage local, par exemple Phong).
  - L'éclairage indirect par d'autres objets : réflexion (objet spéculaire) et réfraction (objet transparent).
  - Une source peut être occultée par un autre objet (ombres portées).

# Contribution indirecte

- ▶ On cherche la couleur du point  $P$  provenant de la réflexion et/ou de la réfraction  $\Rightarrow$  le principe est le même que pour le rayon primaire :
  - Déterminer un/des rayon/s depuis  $P$  (selon les directions de réflexion/réfraction).
  - Pour chaque rayon : calculer l'intersection la plus proche avec la scène pour trouver  $P'$ .
  - Calculer la couleur de  $P'$  (récursivement).
  - La couleur de  $P'$  contribue à la couleur de  $P$  :
    - On affecte généralement un coefficient d'atténuation  $\alpha_P$  à chaque objet pour traduire « plus ou moins » réfléchissant et/ou réfractant.
    - $\text{Couleur}(P) \leftarrow \text{Couleur}(P) + \alpha_P \text{Couleur}(P')$ .



# Rayon d'ombre



- Pour gérer les ombres portées, il suffit de déterminer si une intersection existe entre  $P$  et les sources.
- $\Rightarrow$  rayon (dit d'ombre) d'origine  $P$  et de direction  $PS_i$ .
- Si pas d'intersection alors calculer éclairement direct, sinon éclairement direct=0.

- ▶ On peut se retrouver confronté à une récursivité infinie (exemple : 2 miroirs face à face).
- ▶  $\Rightarrow$  limiter selon une profondeur maximale ou limiter selon le cumul des atténuations (si la contribution devient négligeable : arrêter le calcul).

# Calcul d'éclairément

```
Couleur CalculerCouleur(Rayon d, int profondeur, float attenuation) {
    Si profondeur>PROFONDEUR_MAX ou attenuation<ATTENUATION_MIN alors
        resultat=noir;
    Sinon
        P=Intersection(d,scene); // P doit connaitre son objet (matériel, normale,...)
        Si P=Vide alors resultat=noir;
        Sinon
            couleur=noir;
            // rayons d'ombre et Phong :
            Pour toute source s_i faire
                Si Intersection((PS_i),scene) non Vide Alors
                    couleur = couleur+Phong(P,S_i);
            Fin Si
        Fin Pour
        // rayons secondaires (faire un mélange plus subtil pour éviter la saturation) :
        couleur=couleur+alpha_P_Refracte*
            CalculerCouleur(transmis,profondeur+1,cumul*alpha_P_Refracte);
        couleur=couleur+alpha_P_reflexion*
            CalculerCouleur(transmis,profondeur+1,cumul*alpha_P_Reflexion);
        resultat=couleur;
    Fin Si
}
```



- Il suffit d'itérer pour tous les pixels de l'écran :

```
Pour tout pixel i=(x_i,y_i) faire
  rayon_primaire = droite(eye, i);
  couleur(i)=CalculerCouleur(rayon_primaire,0,1);
  // i.e. profondeur = 0, cumul=1
fin pour
```

Faire attention sur les repères : tous les calculs d'éclairement doivent être faits dans un repère commun (choisir : repère de l'observateur ou repère global de scène).

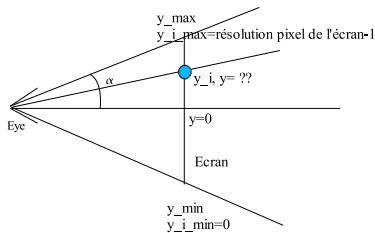
# Rayon et changements de repères

- ▶ On calcule les rayons dans le repère le plus simple possible (exemple : quelle est la définition la plus simple pour l'intersection entre une droite et un cône).
- ▶ Rayon primaire : défini dans le repère Eye puis transformé dans le repère global de scène.
- ▶ On conserve l'équation sous la forme (rayon) :  $P = \text{Origine} + \lambda \text{Directeur}$
- ▶ On transforme alors Origine et Directeur selon les besoins (intersection dans les repères locaux des objets) pour obtenir l'équation dans le repère souhaité.
- ▶  $\Rightarrow$  si  $M_{1 \rightarrow 2}$  est le passage de 1 à 2, et  $P_2 = \text{Origine}_2 + \lambda \text{Directeur}_2$  est le rayon dans le repère 2, alors  $P_1 = M_{1 \rightarrow 2} P_2$  et  $P_1 = M_{1 \rightarrow 2} \text{Origine}_2 + \lambda M_{1 \rightarrow 2} \text{Directeur}_2$  est le rayon dans le repère 1.

# Rayon primaire

- ▶ Origine = Eye, Directeur =  $\overrightarrow{\text{Eye } i}$
- ▶ « Plus proche » signifiera  $\min \lambda$  tels que  $\lambda > 1$  (visible).
- ▶ Propriété pour l'intersection :  $\lambda$  indépendant du repère choisi dans lequel il est calculé (!).

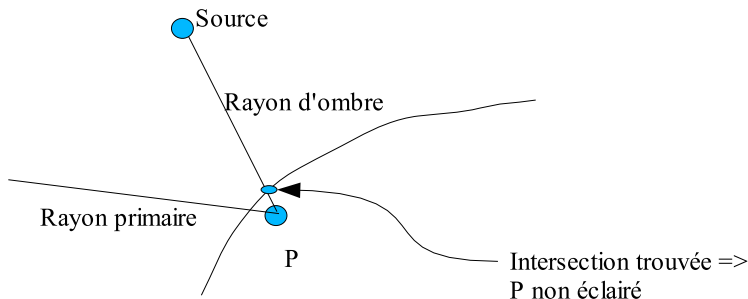
# Exemple : projection perspective



- ▶ Données :  $\alpha$  champ de vision qui intercepte l'écran et  $y_{i_{max}}$  (résolution de l'écran pixel donnée),  $y_i$  : le pixel dont on cherche la couleur.
- ▶ But : trouver  $y$  et  $z$ , coordonnées du pixel dans le repère eye (même raisonnement sur  $x$ ).
- ▶ Soit on connaît  $y_{max}$  (taille de l'écran dans les coordonnées de l'oeil), soit on connaît  $d$  distance de l'écran à l'oeil (repère de l'oeil).
- ▶ Il suffit alors d'utiliser  $z = -d$ ,  $\tan \alpha = \frac{y_{max}}{d}$ ,  $\frac{y}{y_{max}} = \frac{y_i}{y_{i_{max}}}$ .

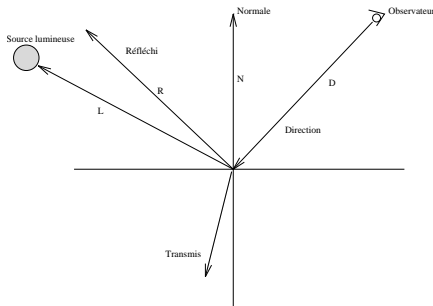
# Rayon d'ombre

- ▶ Origine = P (point d'intersection avec rayon incident), Directeur =  $\overrightarrow{P \text{ Source}}$
- ▶ « Plus proche » signifiera  $\min \lambda$  tel que  $\lambda > 0$ .
- ▶ Aparté : problème d'imprécision numérique classique :



- Résolution par epsilon : problème pour déterminer le bon epsilon.

# Rayons secondaires



- ▶ Origine = P (point d'intersection avec rayon incident), Directeur =  $R$  pour réfléchi ou  $T$  pour réfracté.
- ▶ Réfléchi : à un signe près, nous l'avons déjà vu pour l'éclairément :  $R = -2(D \cdot N)N + D$ .
- ▶ Transmis par réfraction : application de la loi de Descartes :  
$$T = M_1/M_2 * D + [M_1/M_2 * (D \cdot N) - \sqrt{1.0 - (M_1/M_2)^2 * (1.0 - (D \cdot N)^2)}]N$$
- ▶ Au delà d'un angle d'incidence critique, la réfraction devient réflexion (i.e. lorsque  $1.0 - (M_1/M_2)^2 * (1.0 - (D \cdot N)^2) < 0$ ).

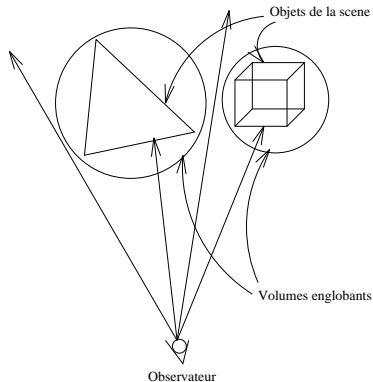
- ▶ Représentation bien adaptée au lancer de rayon (cf algo d'intersection en exercice : changement de repère de la droite dans les feuilles, et tris des  $\lambda$  par fusion).
- ▶ ... à condition que les primitives soit « simples » (calcul d'intersection faisable).

- ▶ Uniquement des intersections de droites avec des plans (et localisation dans les facettes).
- ▶ Il faut prendre garde à éviter les problèmes liés aux erreurs numériques, en tenant compte de la topologie des objets (cas de 2 triangles adjacents par exemple).
- ▶ Calcul de la normale en l'interpolant avec les valeurs aux sommets.
- ▶ Il est rapidement nécessaire de mettre en place des techniques d'accélération pour éviter les nombreux calculs (nombreux polygones).

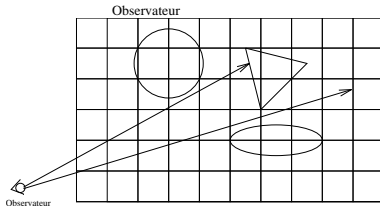


- ▶ Si on obtient une équation polynomiale en  $\lambda$  de degré plus petit que 5, on peut tenter la résolution brutale par radicaux (instabilités numériques peuvent devenir critiques).  
Exemple : surface composées de patches de béziers cubiques  $\Rightarrow$  résolution d'une équation de degré 3.
- ▶ Adopter des méthodes itératives de résolution.
- ▶ Exemple : dichotomie en  $(u, v)$  sur une surface de Bézier en exploitant la propriété d'enveloppe convexe et de normalisation.

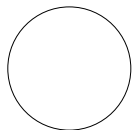
Volumes englobants



Partition de l'espace



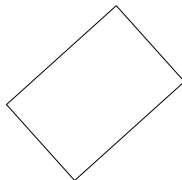
# Volumes englobants



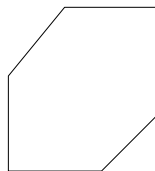
Sphère



AABB  
(Axis Aligned Bounding Box)

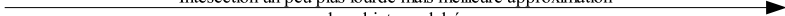


OBB  
(Oriented Bounding Box)

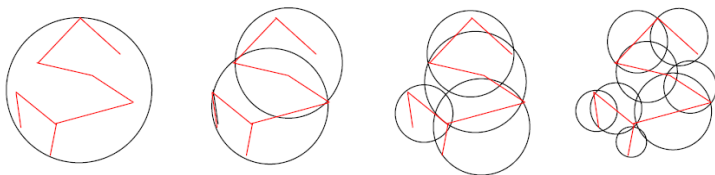


K-dop (k=3 ici)  
(discrete oriented polytope)

Intesection un peu plus lourde mais meilleure approximation  
des objets englobés

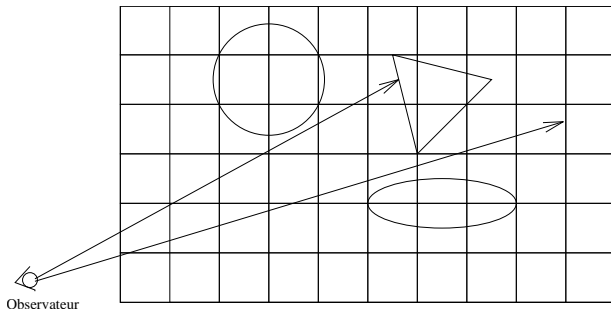


# Hierarchie de volumes englobants



⇒ « descendre » dans les niveaux de précision si une intersection avec une sphère est trouvée.  
Principal problème : construire la hiérarchie automatiquement.

# Partition de l'espace



- ▶ Le rayon « avance » de cube en cube (lancer de rayon dit discret : parcours du rayon incrémentalement)
- ▶ Si un objet se trouve dans un cube :
  - ou bien on effectue l'intersection avec l'objet.
  - ou bien on augmente en résolution (éventuellement hiérarchiquement).
- ▶ Principal problème : énumération des objets dans les cubes.