Le lancé de rayons:
Du modèle 3D à l'image photoréaliste

Contexte

Rendu: Modèle de données 3D $\longrightarrow$ **Image**

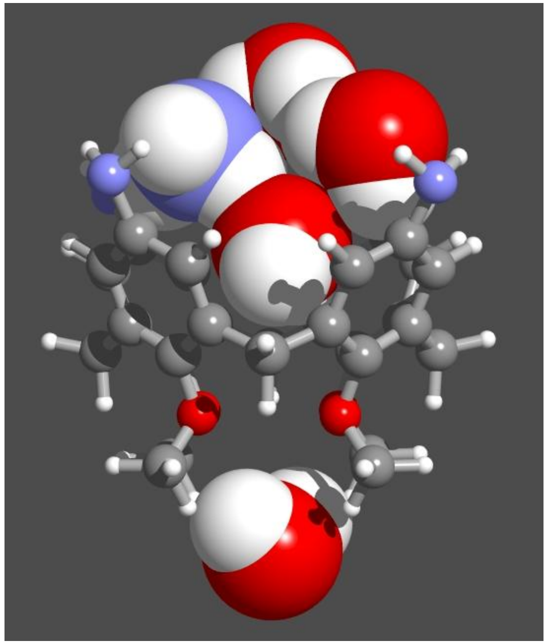
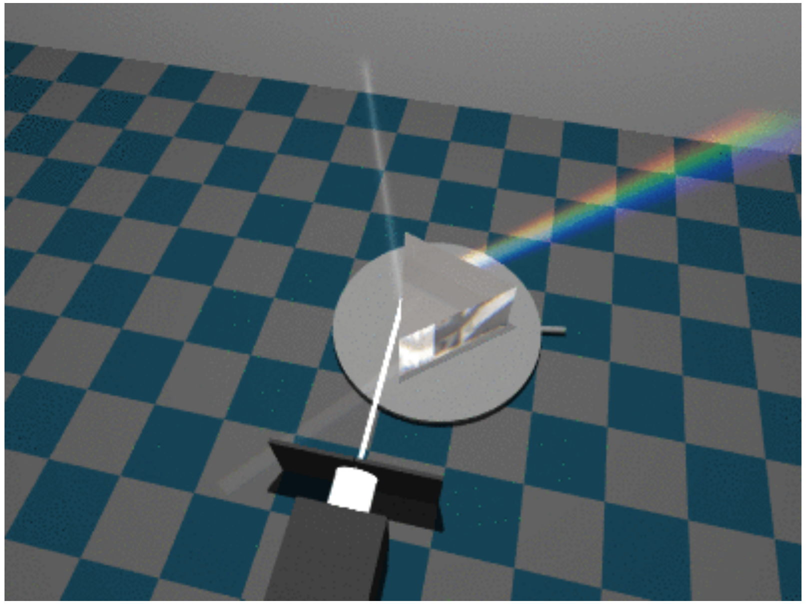
Ex. Modèle 3D sphère:

$$x^2 + y^2 + z^2 = 1 \xrightarrow{?}$$

Image

Contexte

Exemple d'application:

Contexte

Rendu: Modèle de données 3D ⟶ **Image**

Ex. Modèle 3D sphère:

$$x^2 + y^2 + z^2 = 1 \xrightarrow{?}$$

Image

Plusieurs solutions
(**+**/**-**)

- Rendu projectif
- Reye
- Lancé de rayons (*ray tracing*)

Plan de la présentation

1/ Principe général du lancé de rayons

2/ Cas d'application sur une scène simple
=> *Voir des objets*

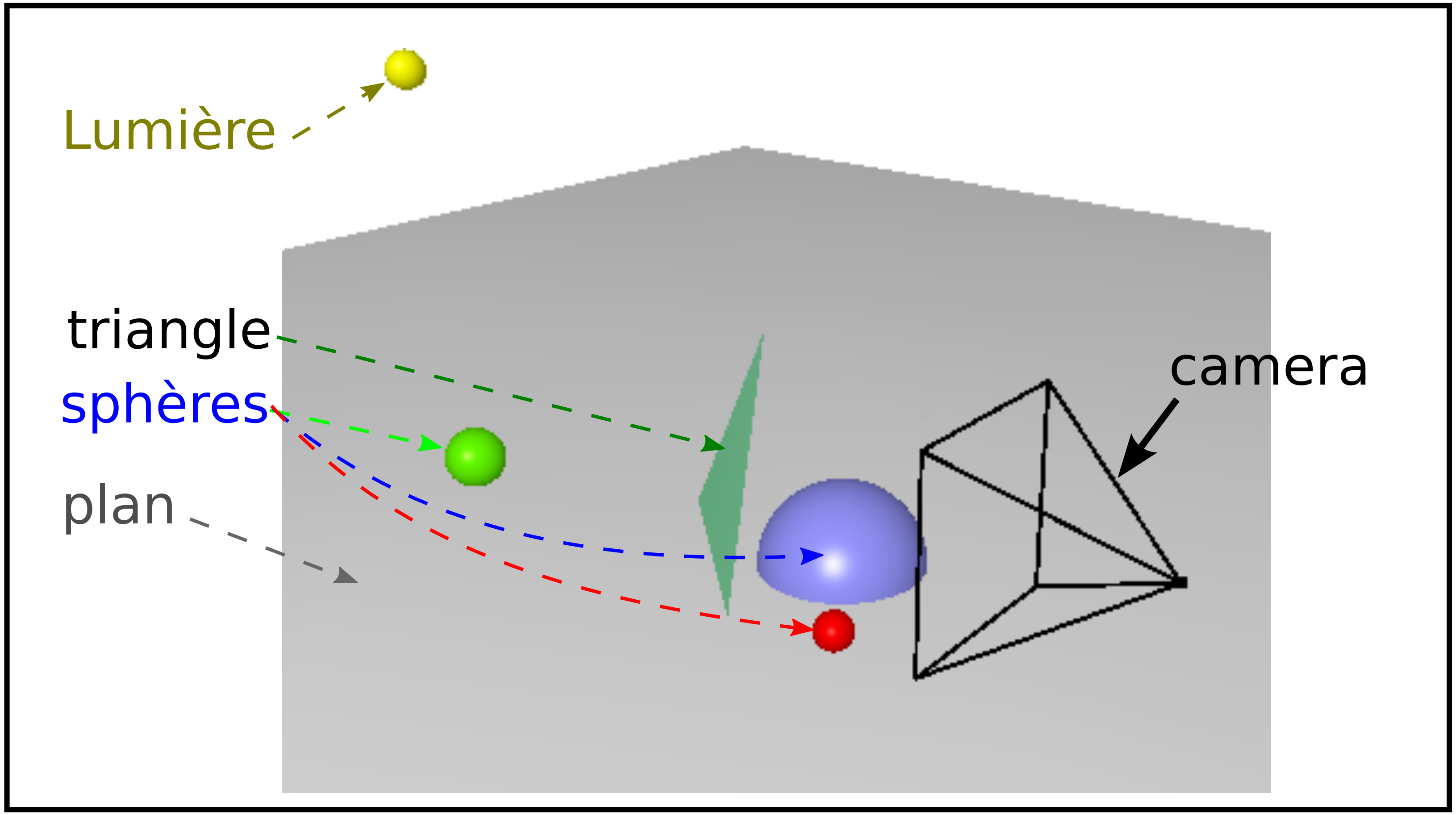
3/ L'illumination
=> *Aspect visuel plaisant*

4/ Modèle physique
=> *Vers le photoréalisme*

1/ Principe général du lancé de rayons

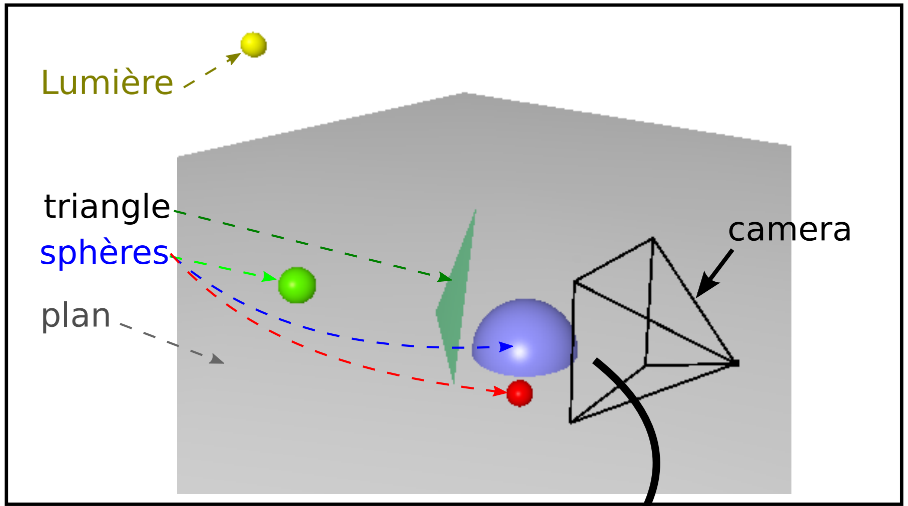
Contexte: Scène 3D

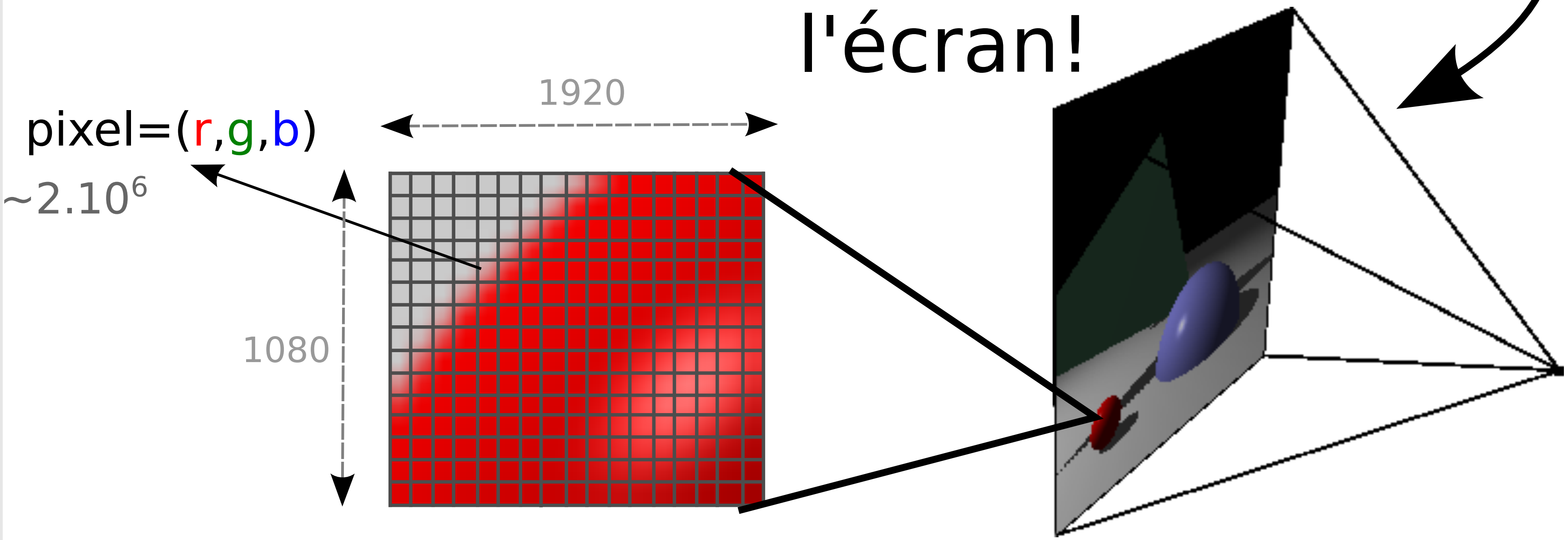
Scène 3D=
- Objet(s) 3D
  - Géométrie: équation
  - Couleur/matériaux
- Caméra/écran
- Source(s) de lumière



Contexte: Scène 3D

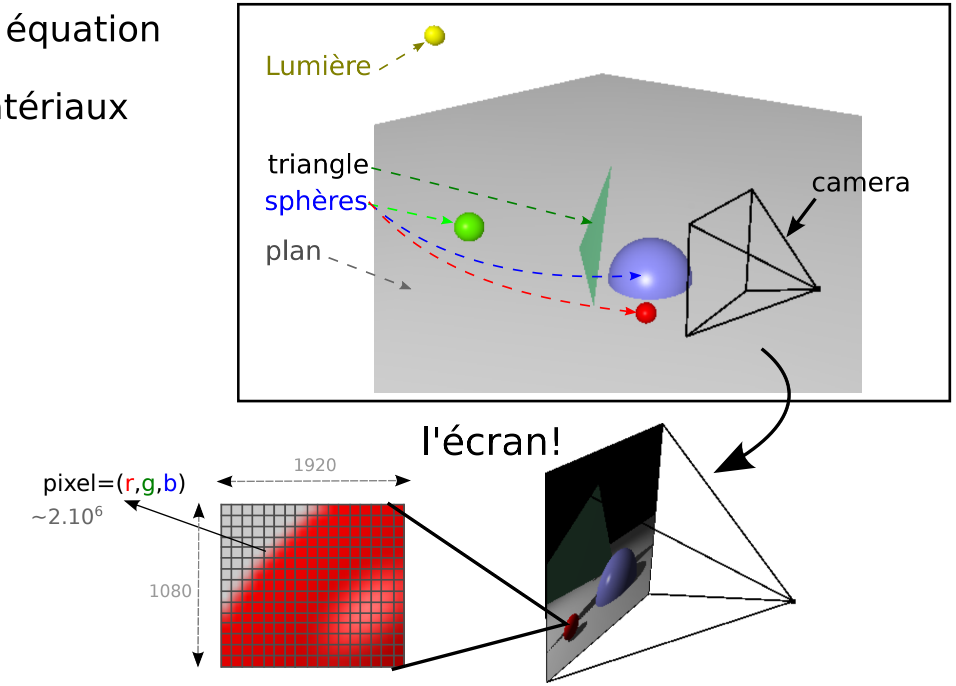
Scène 3D=
- Objet(s) 3D
  - Géométrie: équation
  - Couleur/matériaux
- Caméra/écran
- Source(s) de lumière





Contexte: Scène 3D

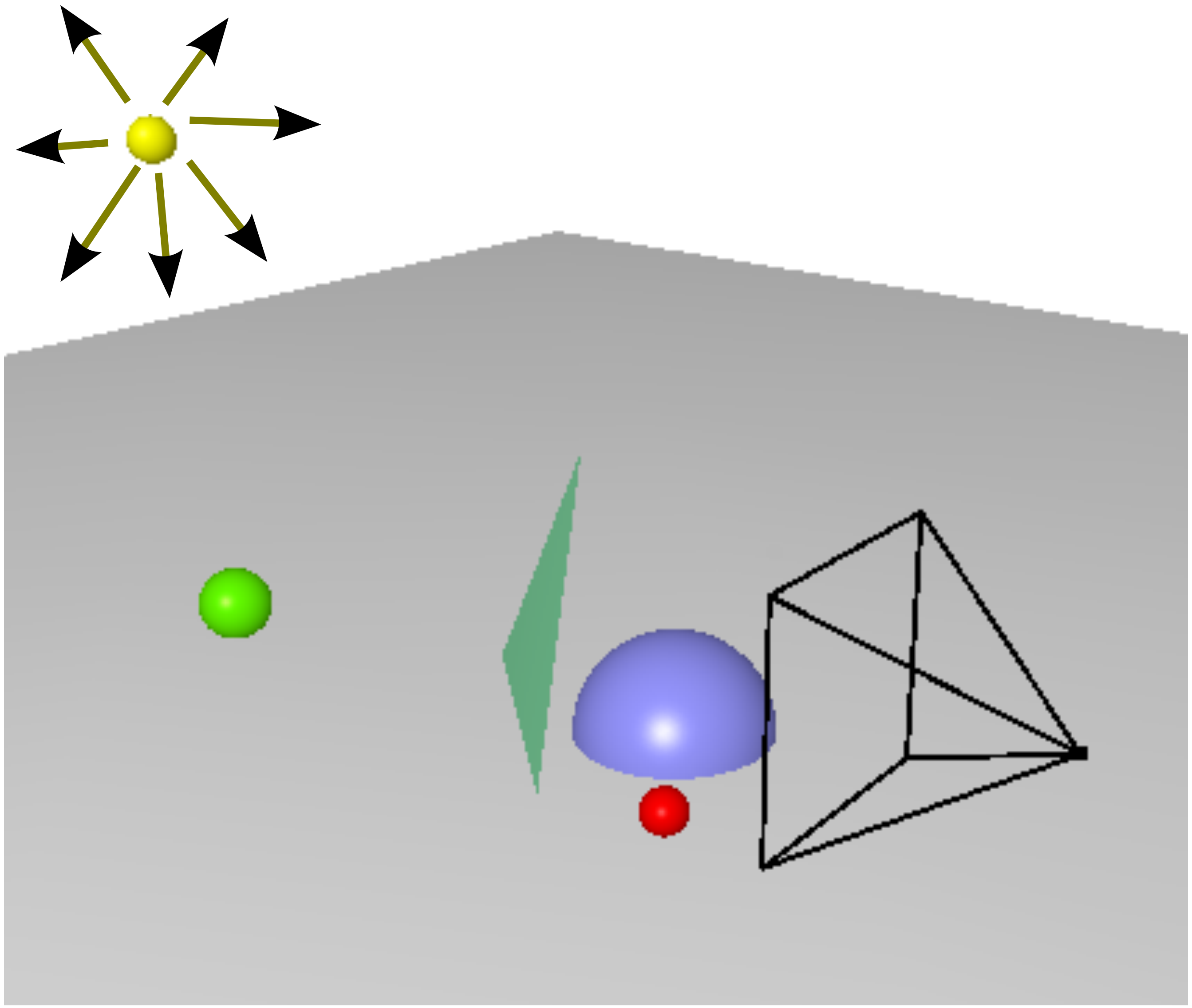
Scène 3D=
- Objet(s) 3D
  - Géométrie: équation
  - Couleur/matériaux
- Caméra/écran
- Source(s) de lumière



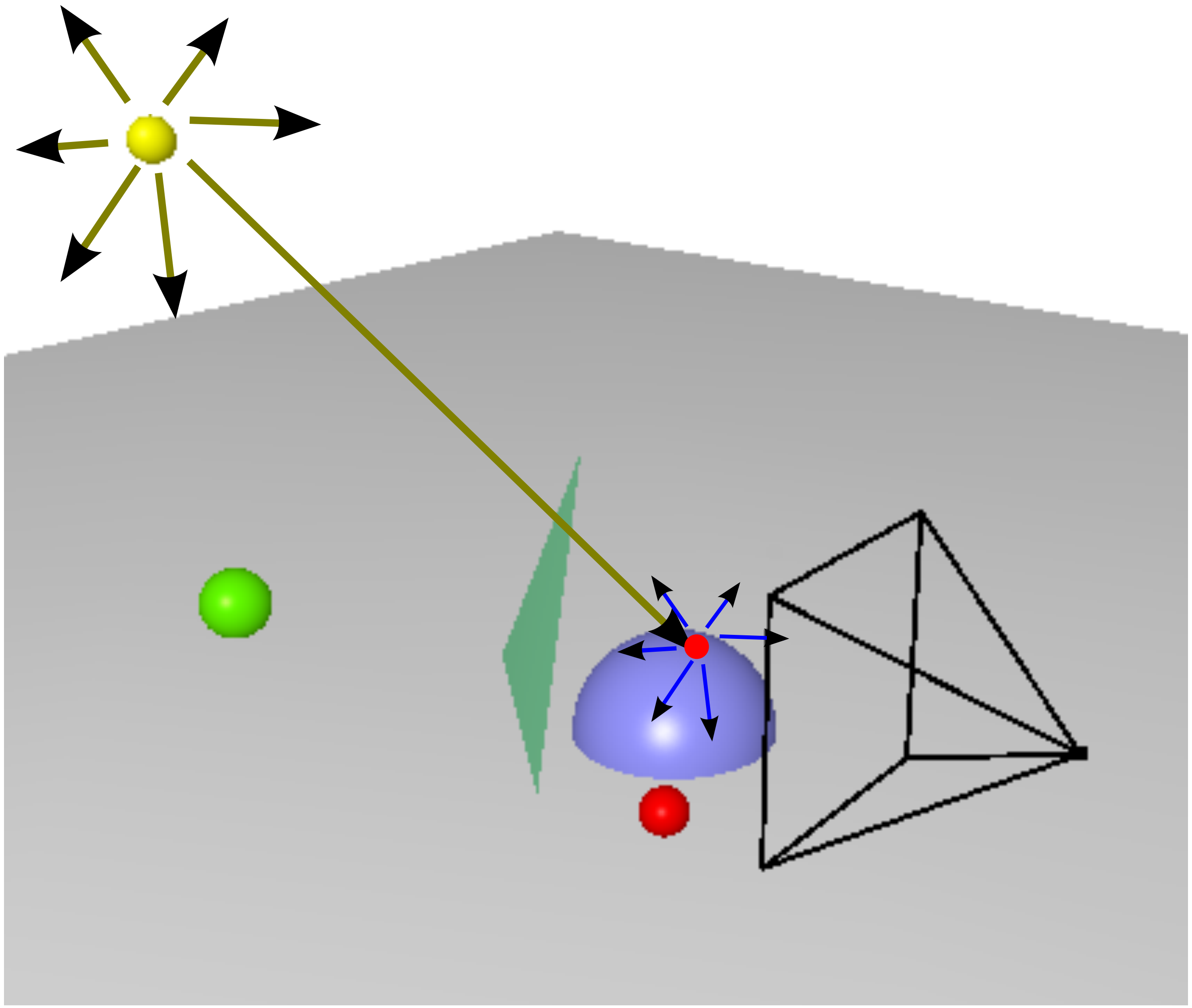
Question:

Quelles couleurs affecter aux pixels ?

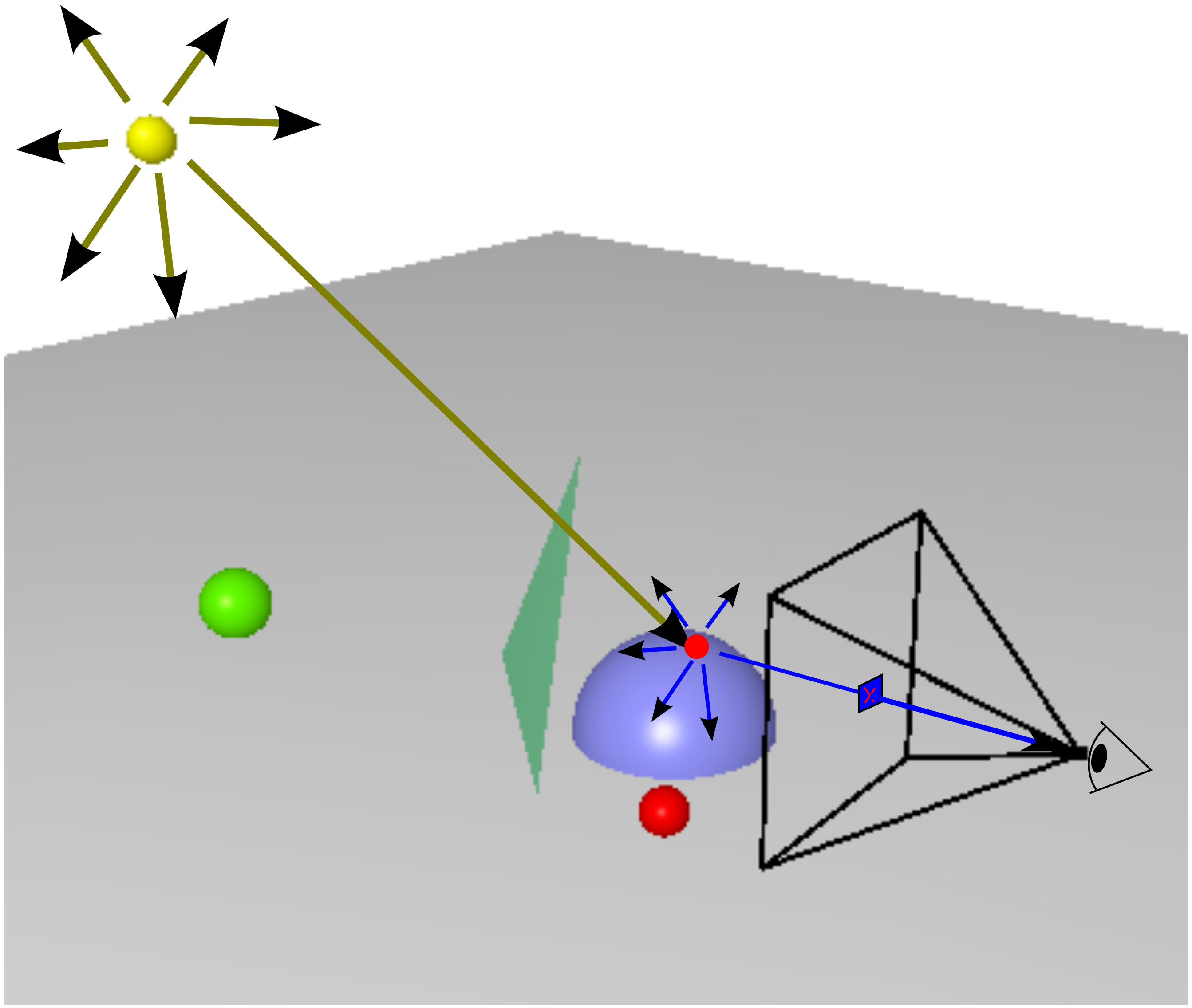
Dans la réalité?



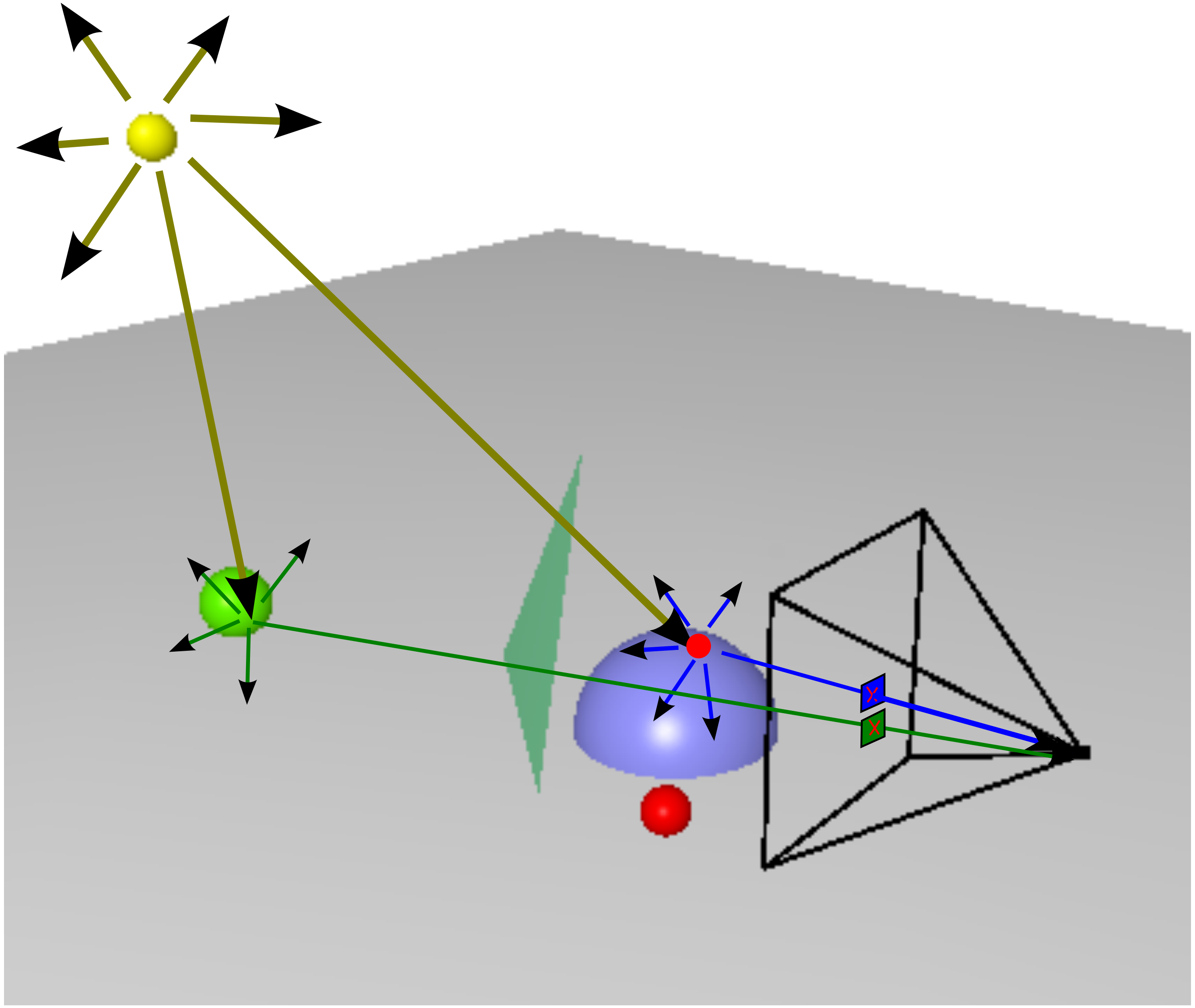
Dans la réalité?



Dans la réalité?



Notre 1er modèle



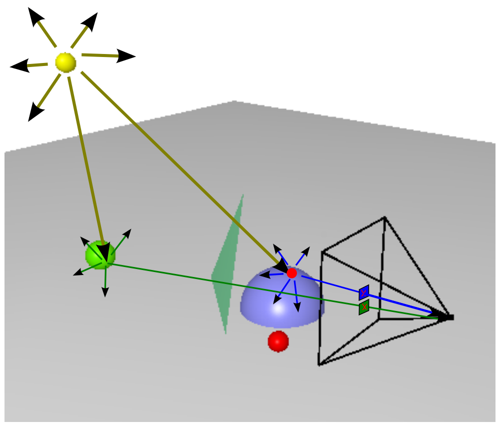
Pour *toutes sources lumiere*
  Pour *toutes directions* ***d1***
    Lance_rayon(***d1***)
  Fin Pour
Fin Pour

Lance_rayon(**d**)
{
  Si d *intersecte objet*
    Sauve couleur
    Pour *toutes directions* ***d2***
      Lance_rayon(**d2**)
    Fin Pour
  Fin Si

  Si d *intersecte ecran*
    Affecte couleur courante
  Fin Si

  Exit
}

Notre 1er modèle

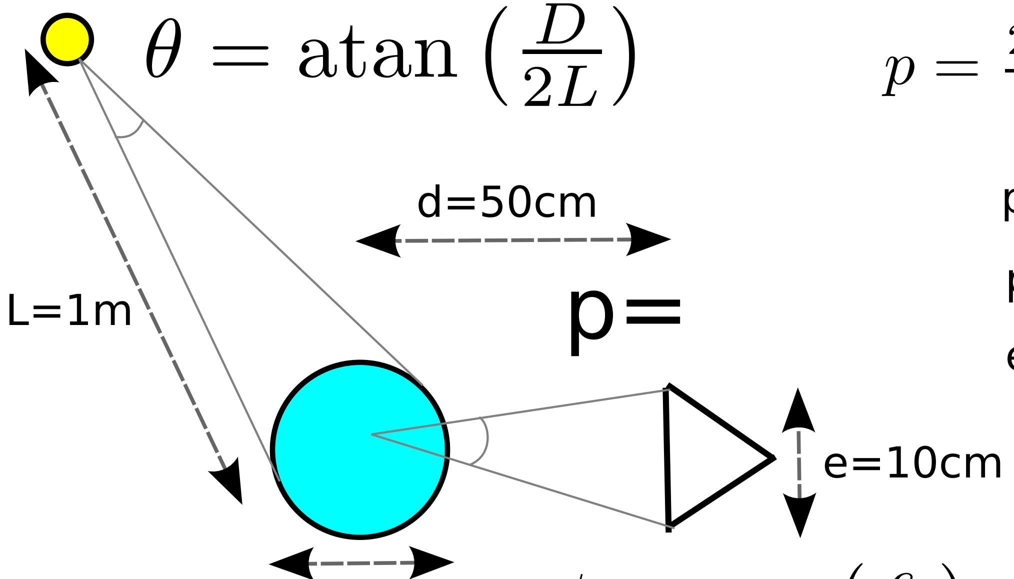


+ Conceptuellement simple
+ Physiquement ok

- Complexité algorithmique

  → Recursions infinies

ex.



$$\theta = \operatorname{atan}\left(\frac{D}{2L}\right)$$

$$\phi = \operatorname{atan}\left(\frac{e}{2d}\right)$$

$$p = \frac{2\pi(1-\cos(\theta))}{4\pi} \times \frac{2\pi(1-\cos(\phi))}{4\pi}$$

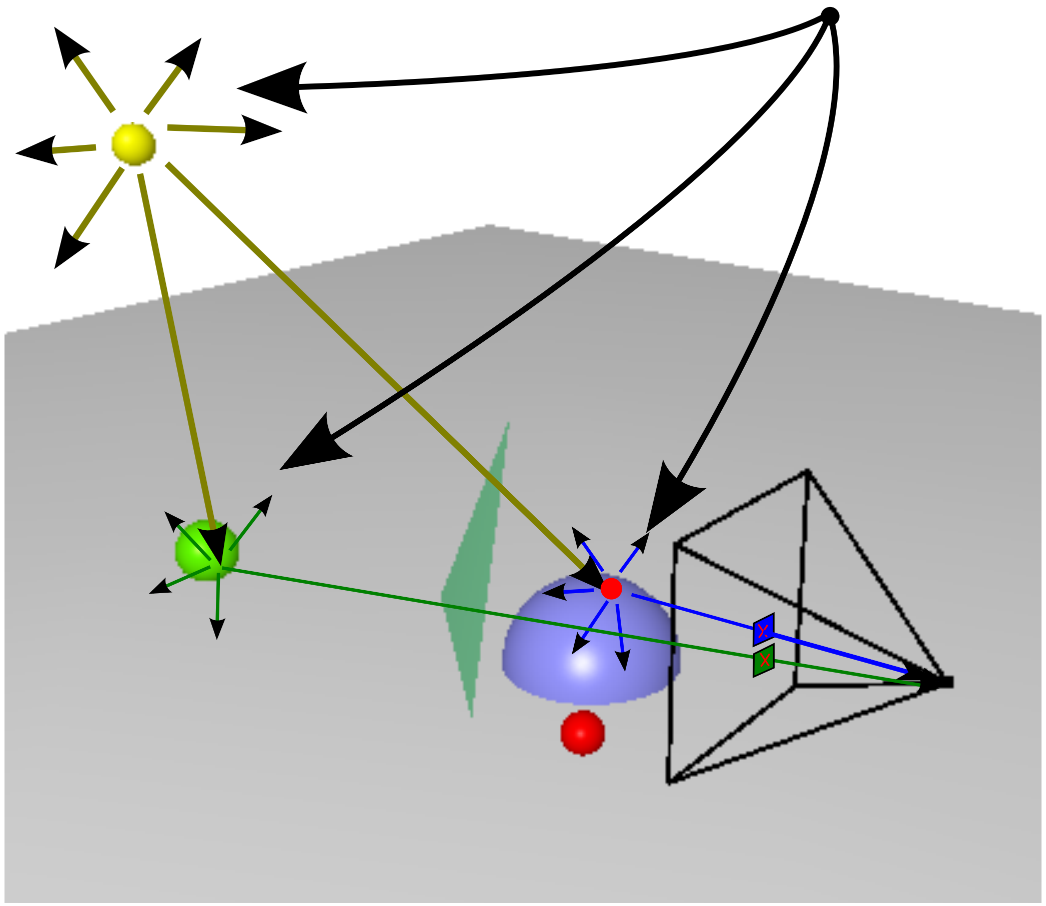
p touche=0.0015%

p pixel=0.0000000007%

envoyer: 130 000 000 000 rayons

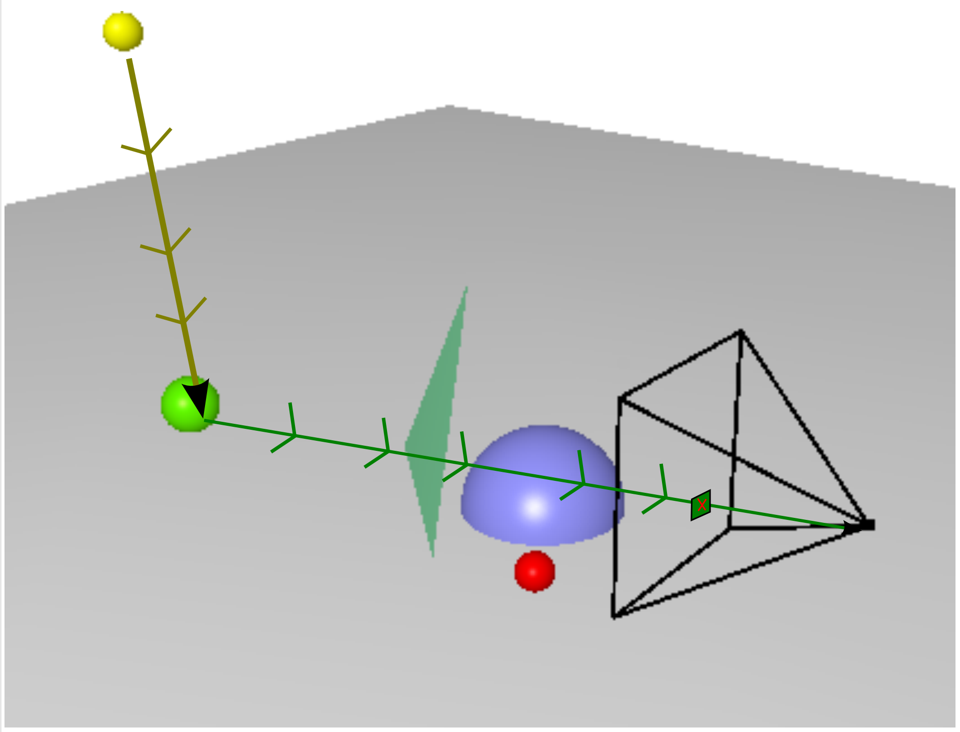
Acceleration

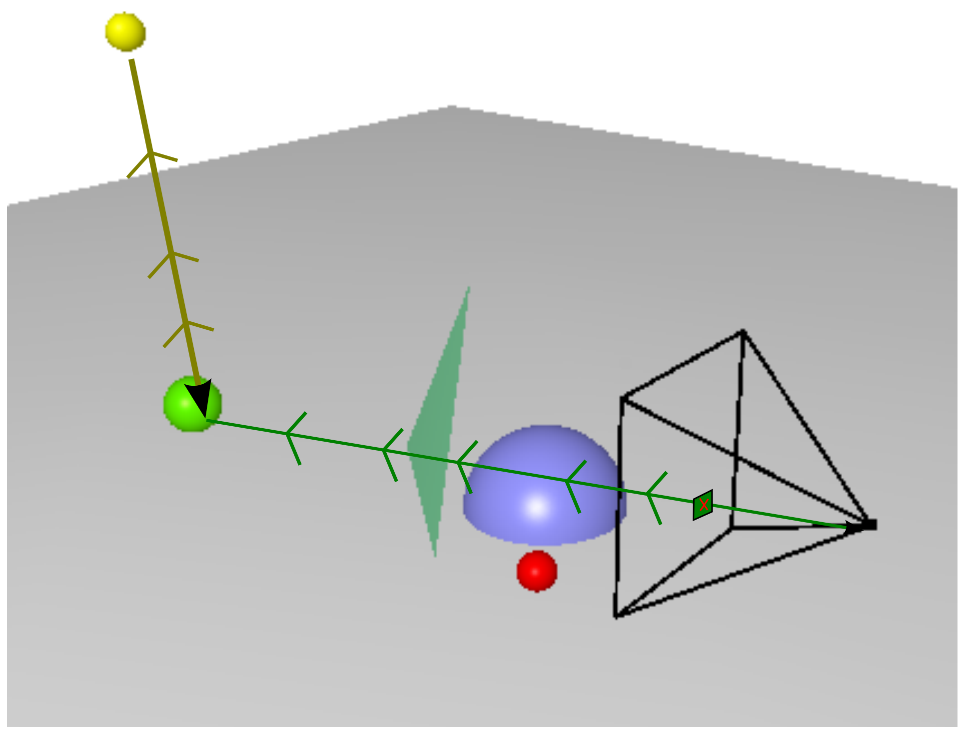
Problème = rayons inutiles



Acceleration

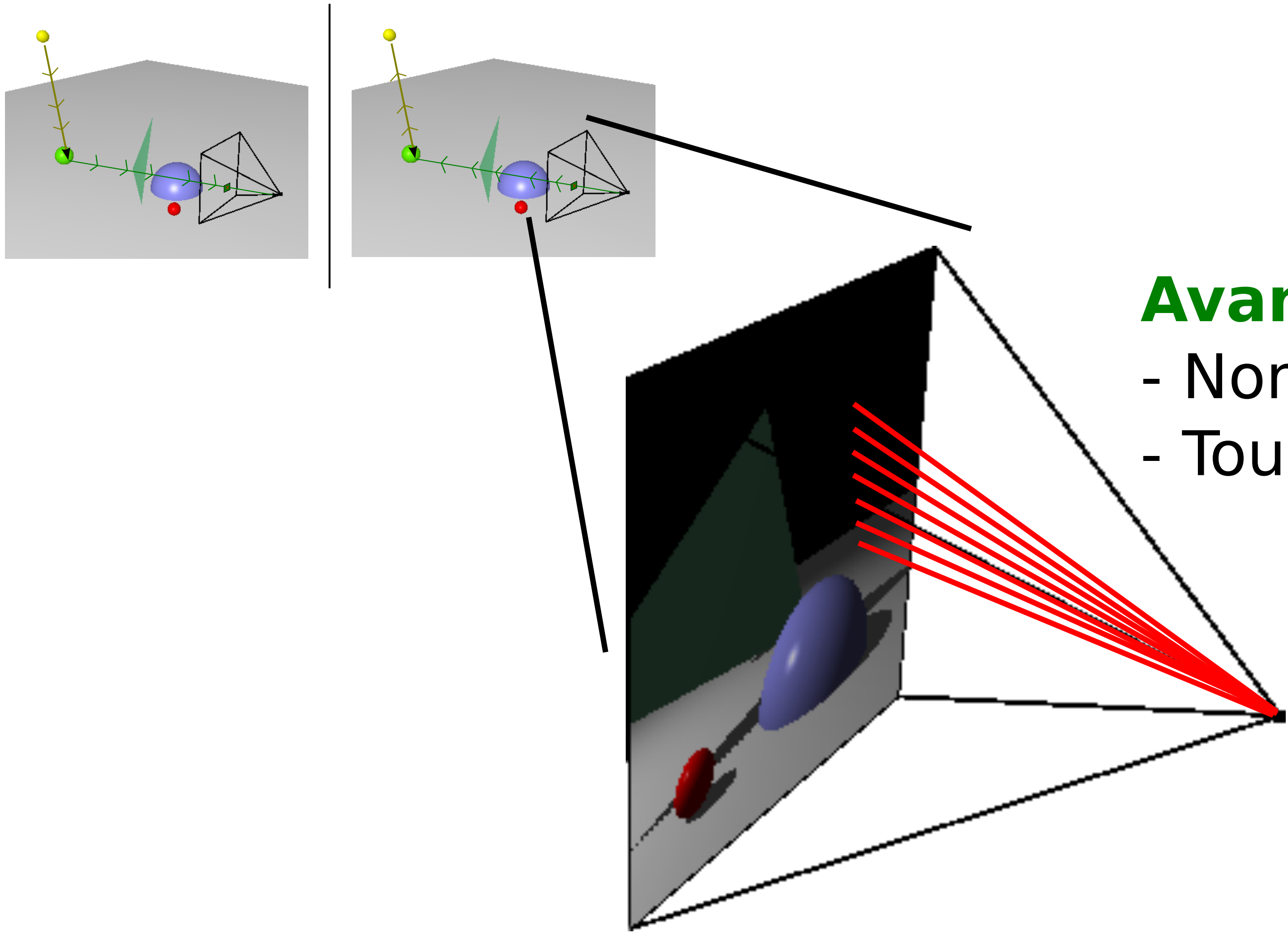
Principe du retour inverse de la lumière





Acceleration

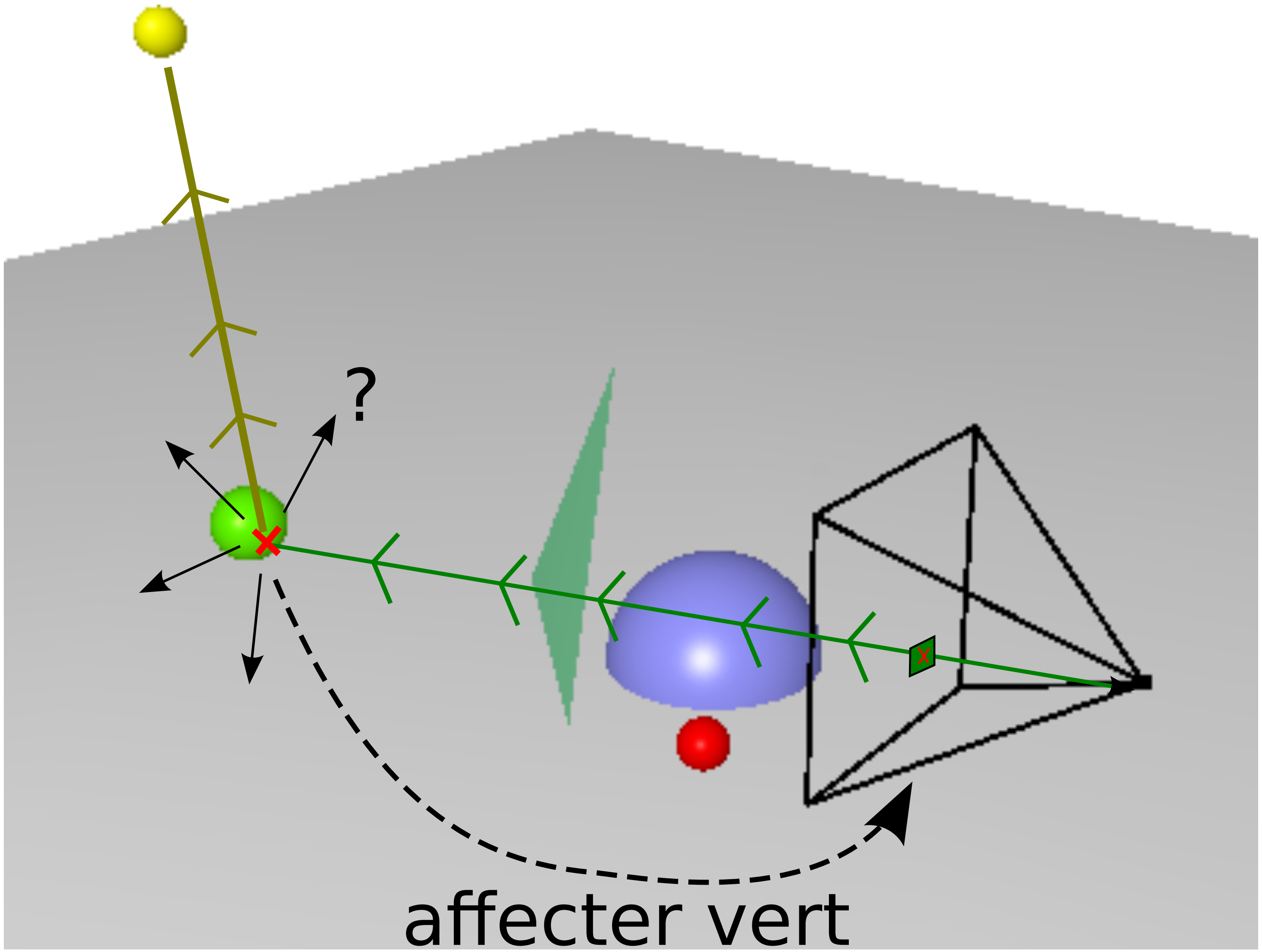
Principe du retour inverse de la lumière



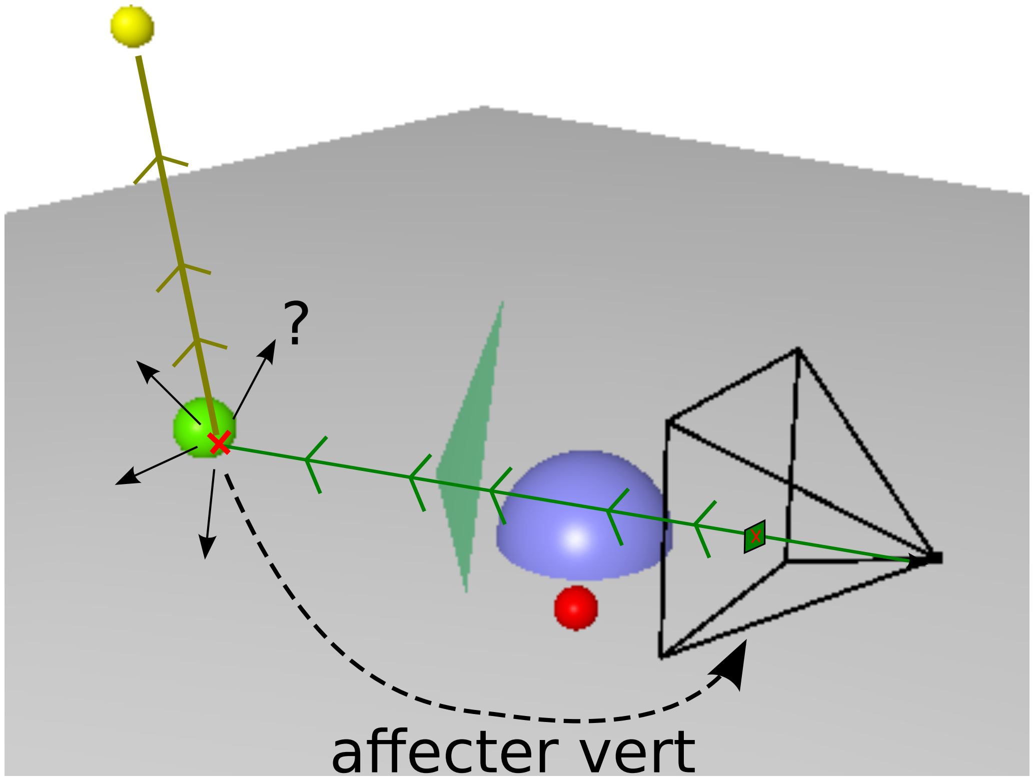
Avantage:
- Nombre fini de rayons
- Tous utiles

Acceleration

Principe du retour inverse de la lumière



Acceleration



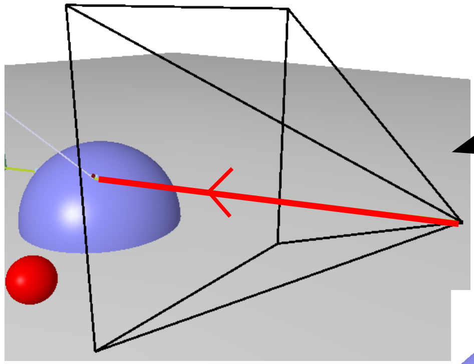
1/ Principe du retour inverse de la lumière
+ physique OK

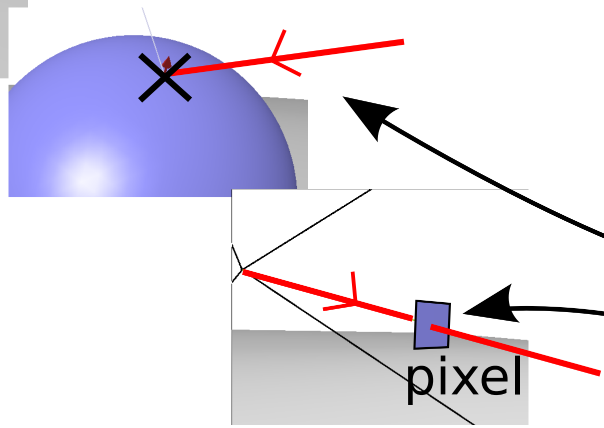
2/ Pas de diffusion secondaire
- physique perdu

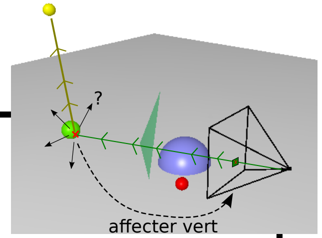
quantification:
~2 000 000 rayons

AN: 0.1ms/rayons :
3min/image
vs 150jours/image

Recapitulatif



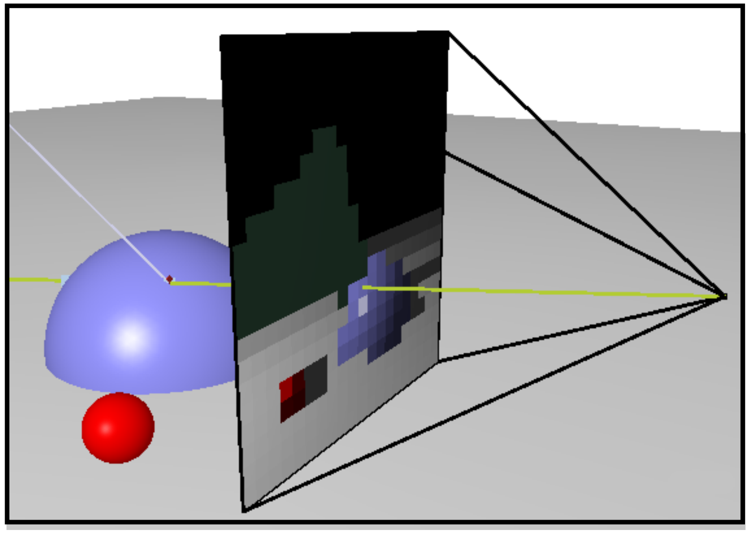




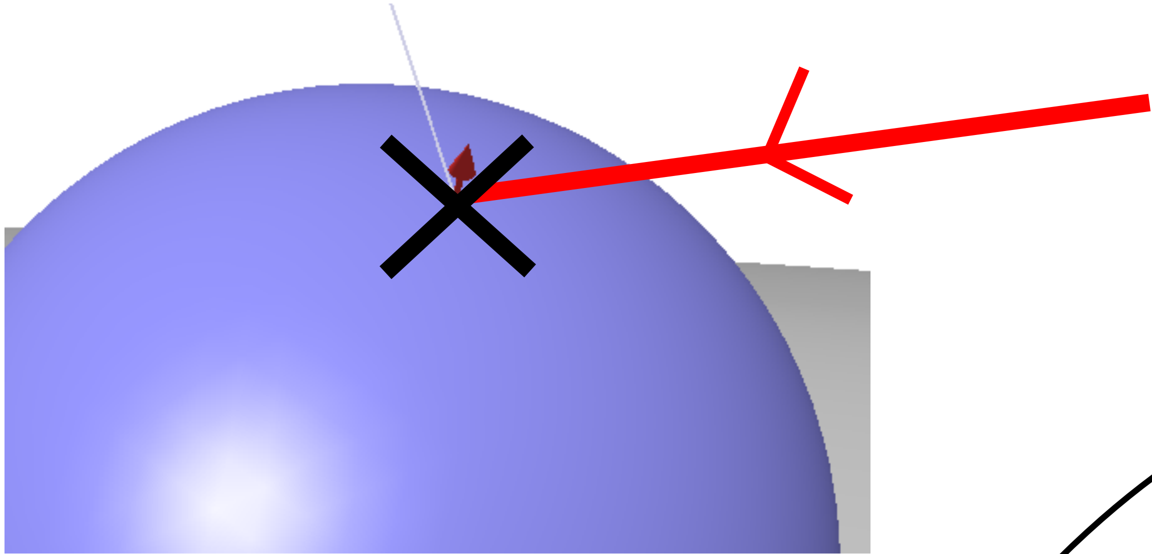
Pour k=1..N_pixels
Lance_rayon(**d**$_k$)
Fin Pour

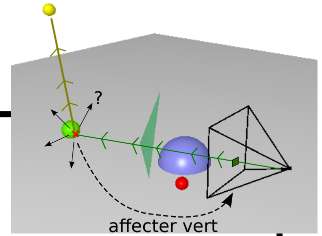
Lance_rayon(**d**)
{
Si d *intersecte objet*
Affecte couleur objet au pixel
Fin Si

Exit
}



Recapitulatif





Pour k=1..**N_pixels**
  Lance_rayon(**d**$_k$)
Fin Pour

Lance_rayon(**d**)
{
Si d *intersecte objet*
  Affecte couleur objet au pixel
Fin Si

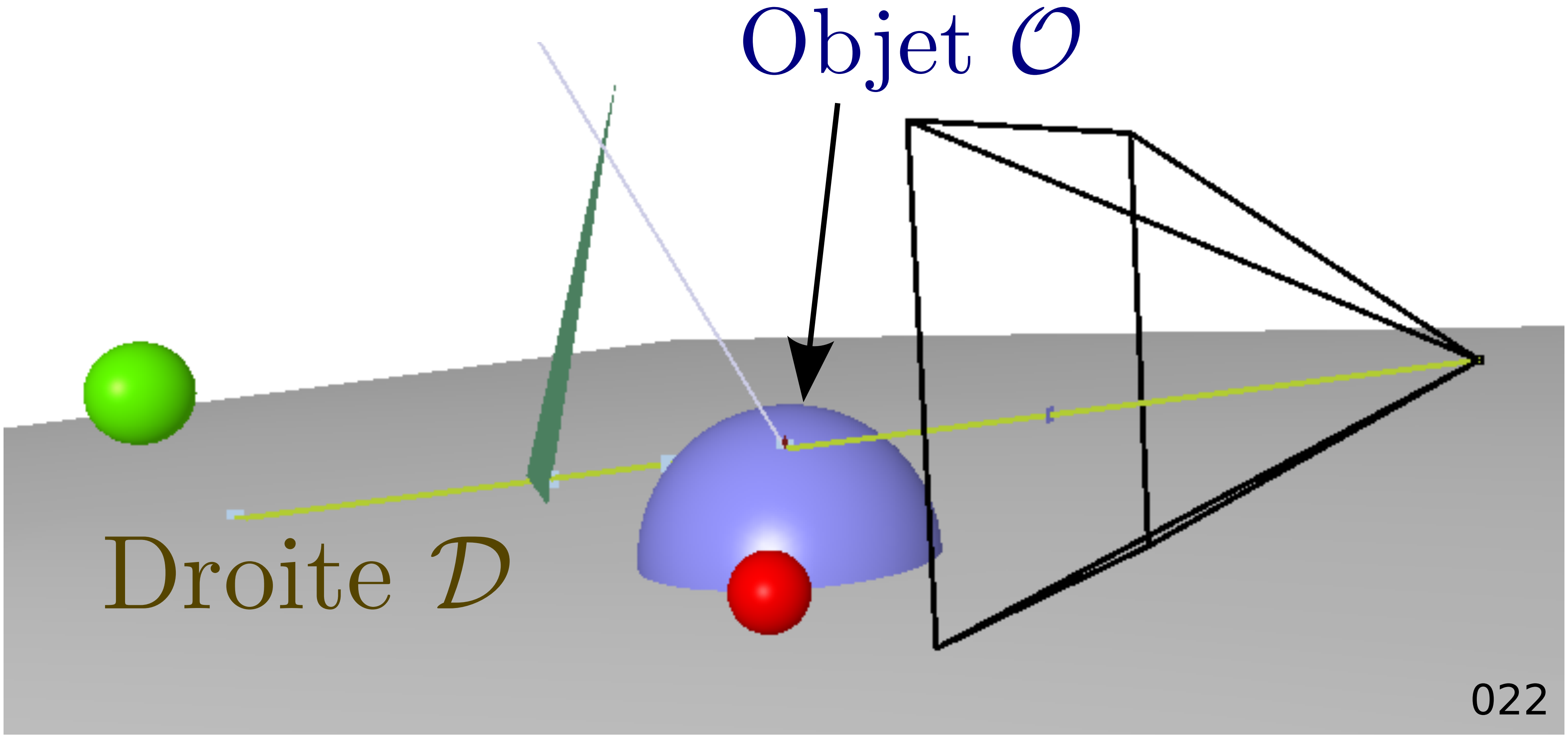
Exit
}

Complexité

Formalisation

Pour toutes droites $\mathcal{D}$

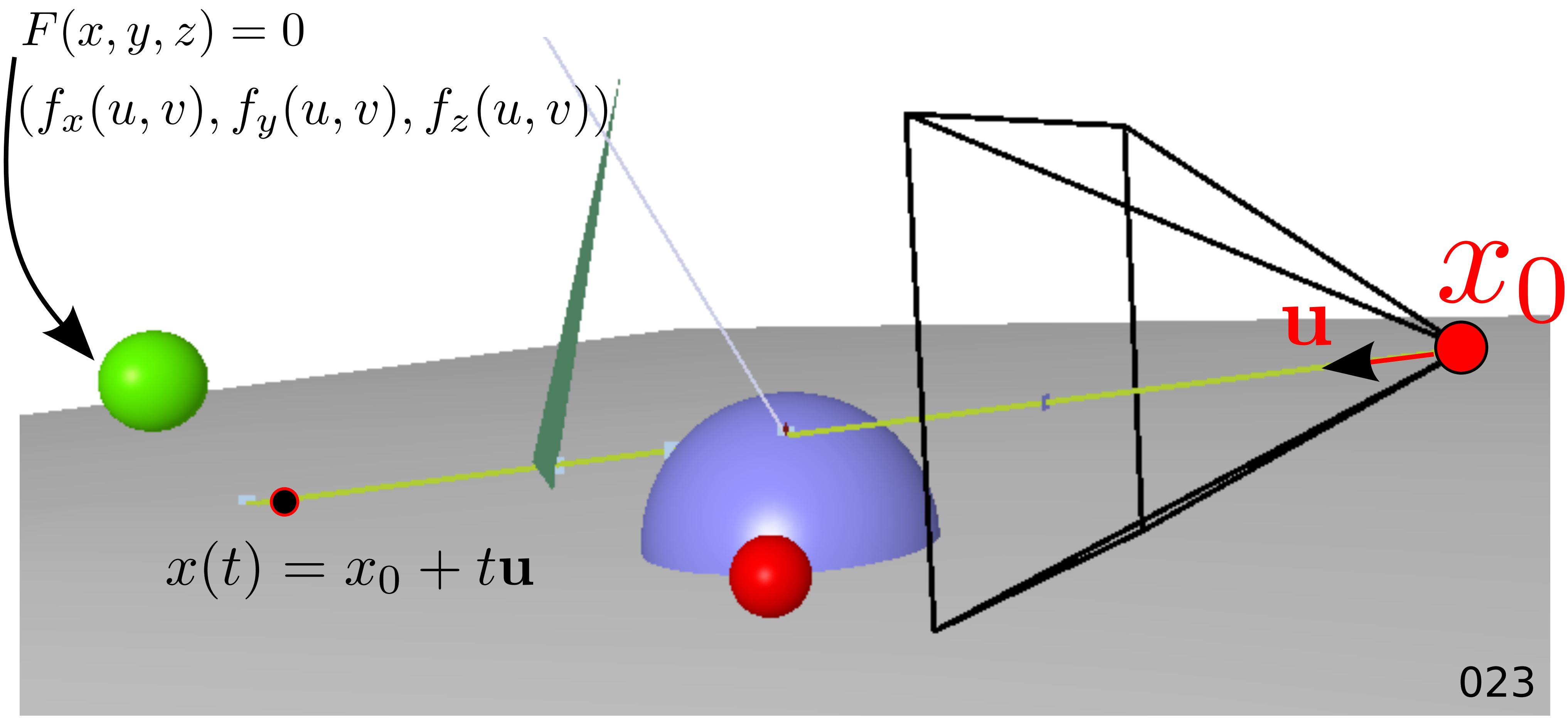
Connaitre $\mathcal{D} \cap \mathcal{O}$



Formalisation

Pour toutes droites $\mathcal{D}$

$$\text{Connaitre } \mathcal{D} \cap \mathcal{O}$$



Formalisation

1: On cherche

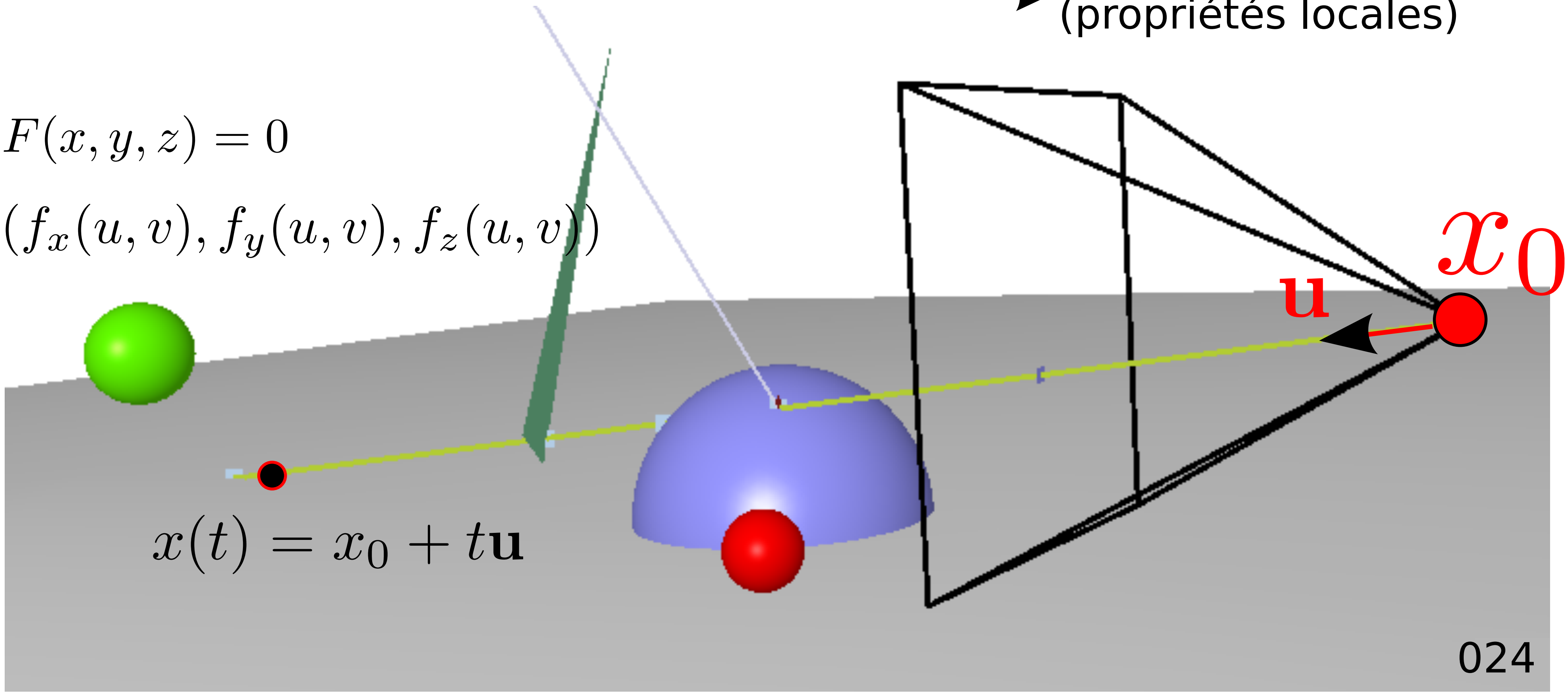
$$\begin{cases} F(x,y,z) = 0 \\ x(t) = x_0 + t\mathbf{u} \end{cases}$$

2: On résout pour t

On en déduit:
x(t) et **n**(t)

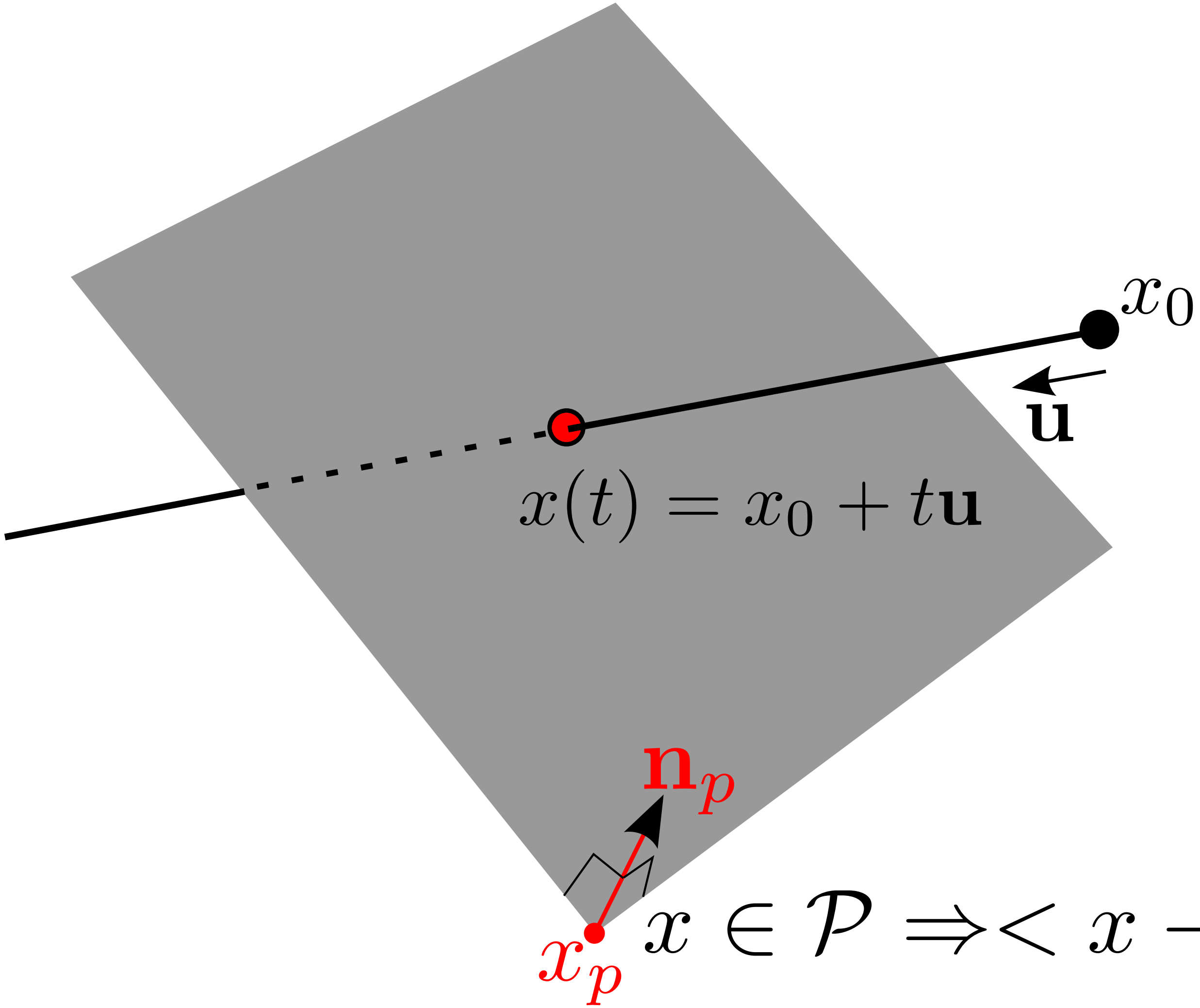
On garde t>0

Illumination
(propriétés locales)



2/ Cas d'application sur des scènes simples

Cas du plan



$$x \in \mathcal{P} \Rightarrow < x - x_p, \mathbf{n}_p >= 0$$

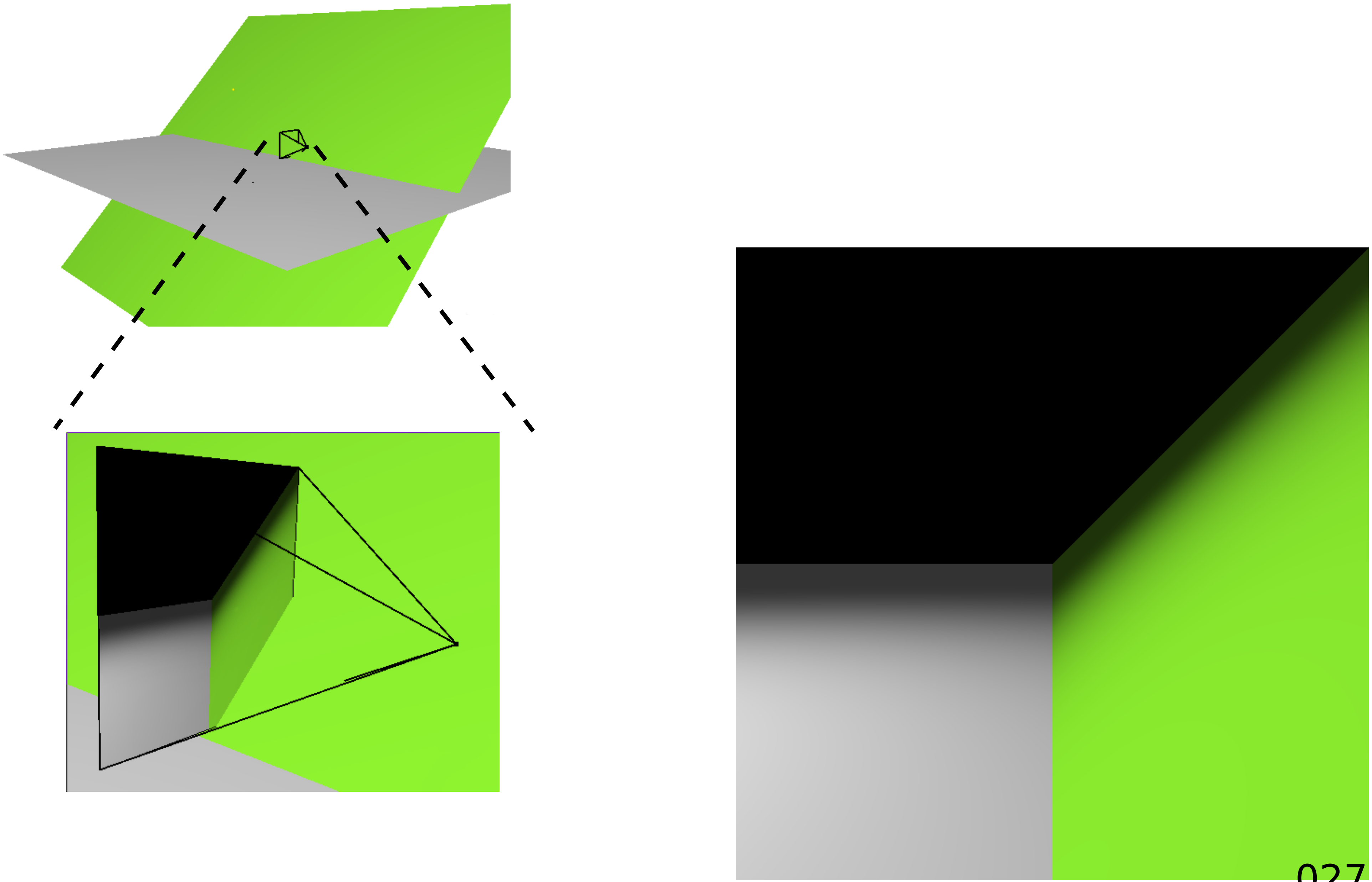
Il faut résoudre:

$$\begin{cases} x(t) = x_0 + t\mathbf{u} \\ < x(t) - x_p, \mathbf{n}_p >= 0 \end{cases}$$

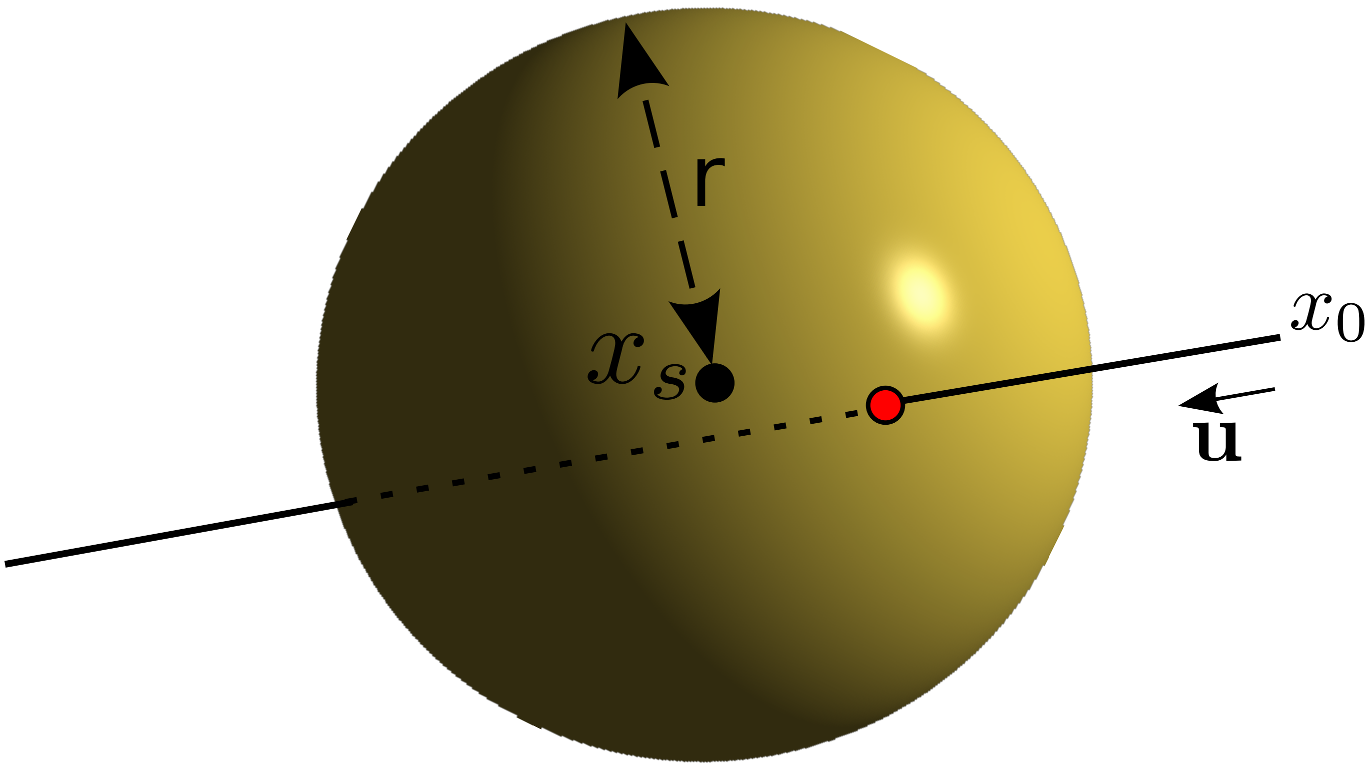
2 cas

- Coplanaire → 0 intersection
- Intersection
  - t>0 → 1 intersection
  - t<0 → 0 intersection

Cas du plan

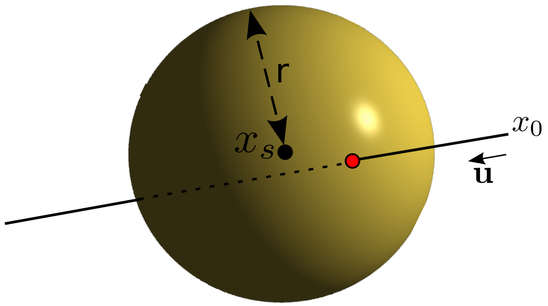


Cas de la sphère



$$x \in \mathcal{S} \Rightarrow \|x - x_s\| = r$$

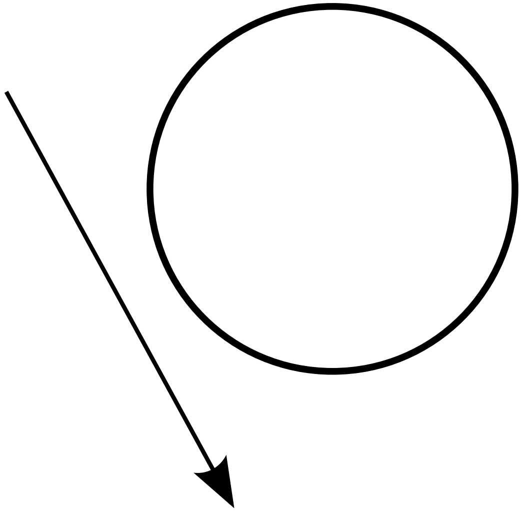
Cas de la sphère



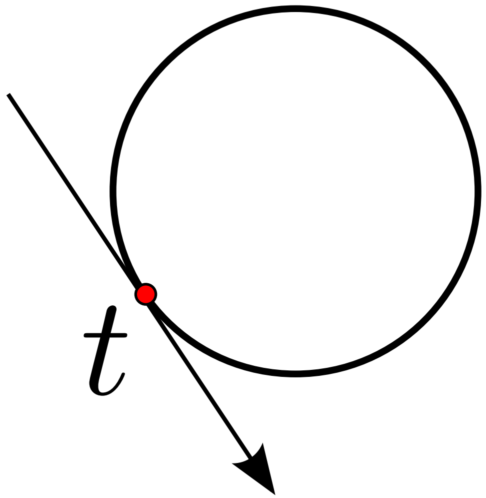
$x \in \mathcal{S} \Rightarrow \|x - x_s\| = r$

$$\begin{cases} x(t) = x_0 + t\mathbf{u} \\ \|x(t) - x_s\| = r \end{cases}$$

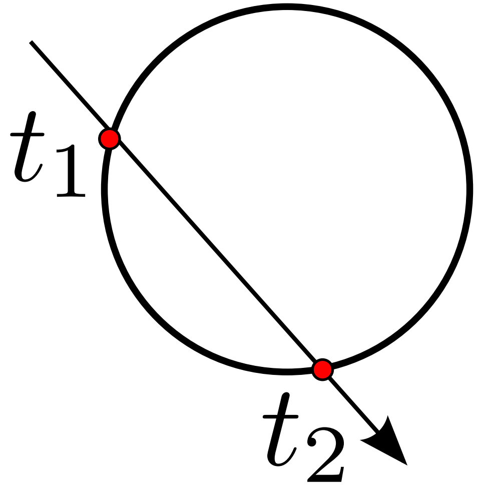
Cas 1:



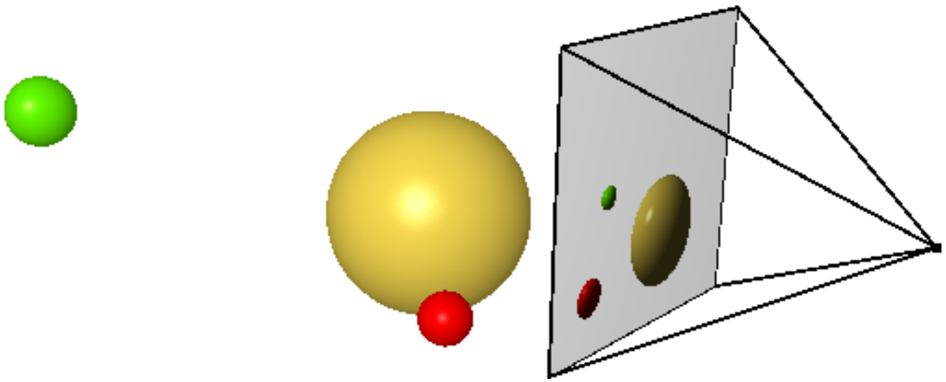
Cas 2:

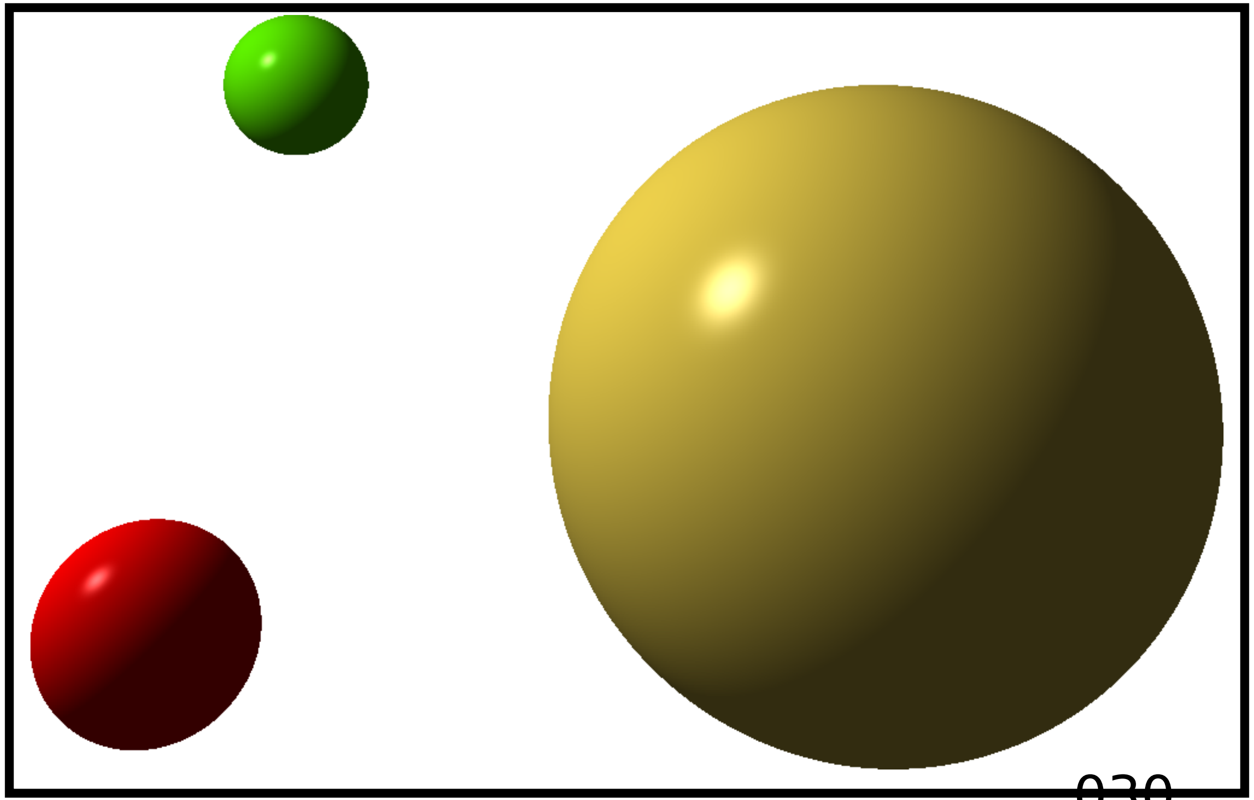


Cas 3:



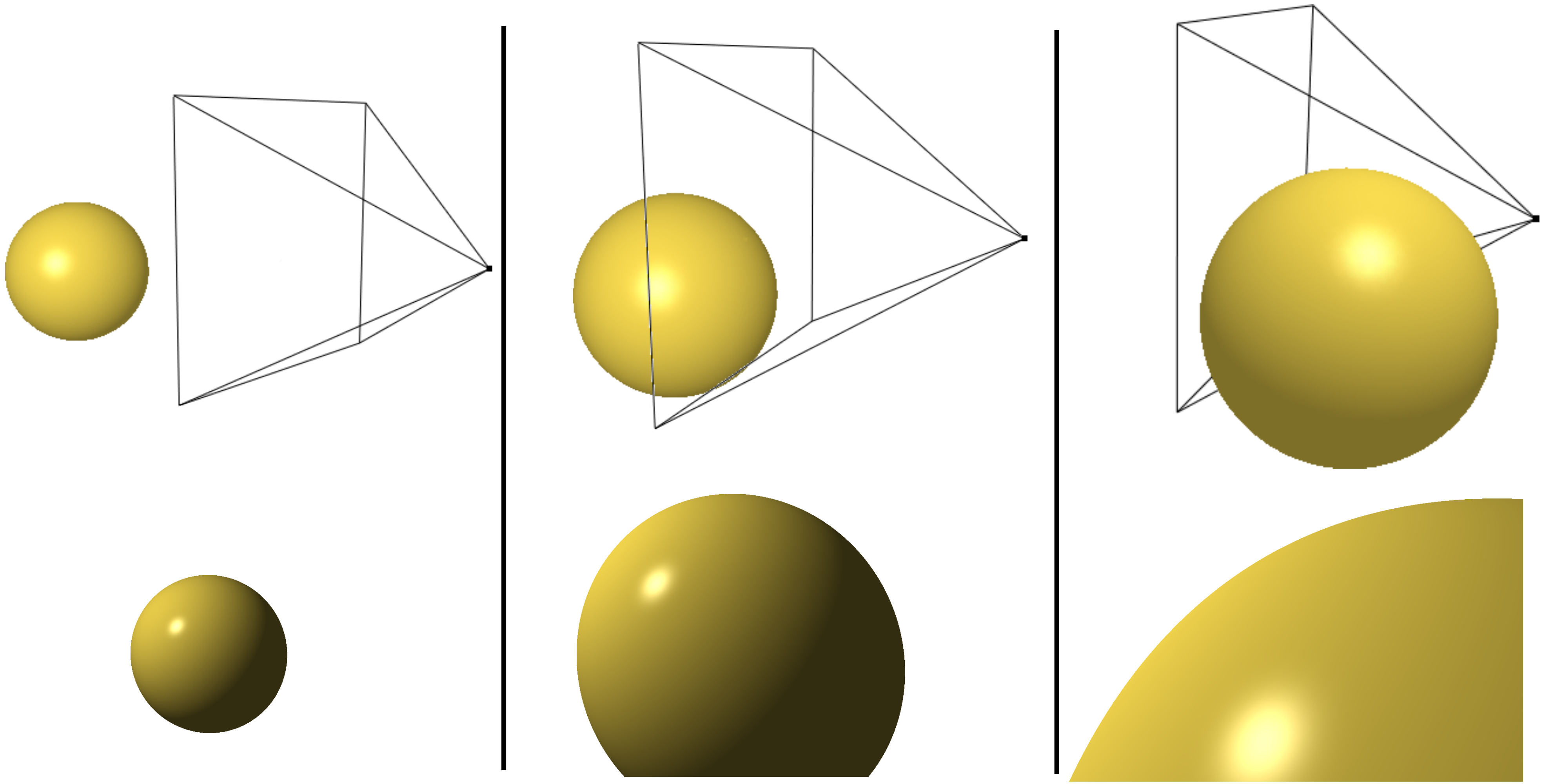
Cas de la sphère





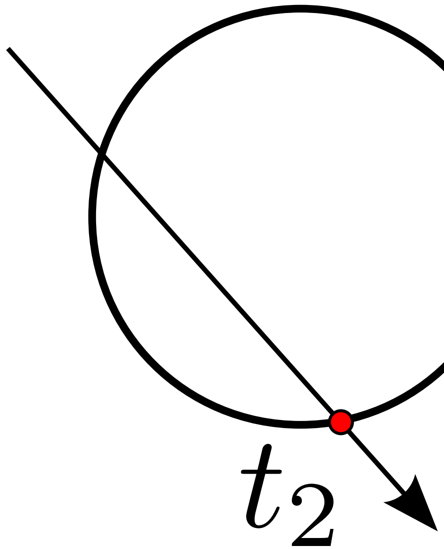
Cas de la sphère

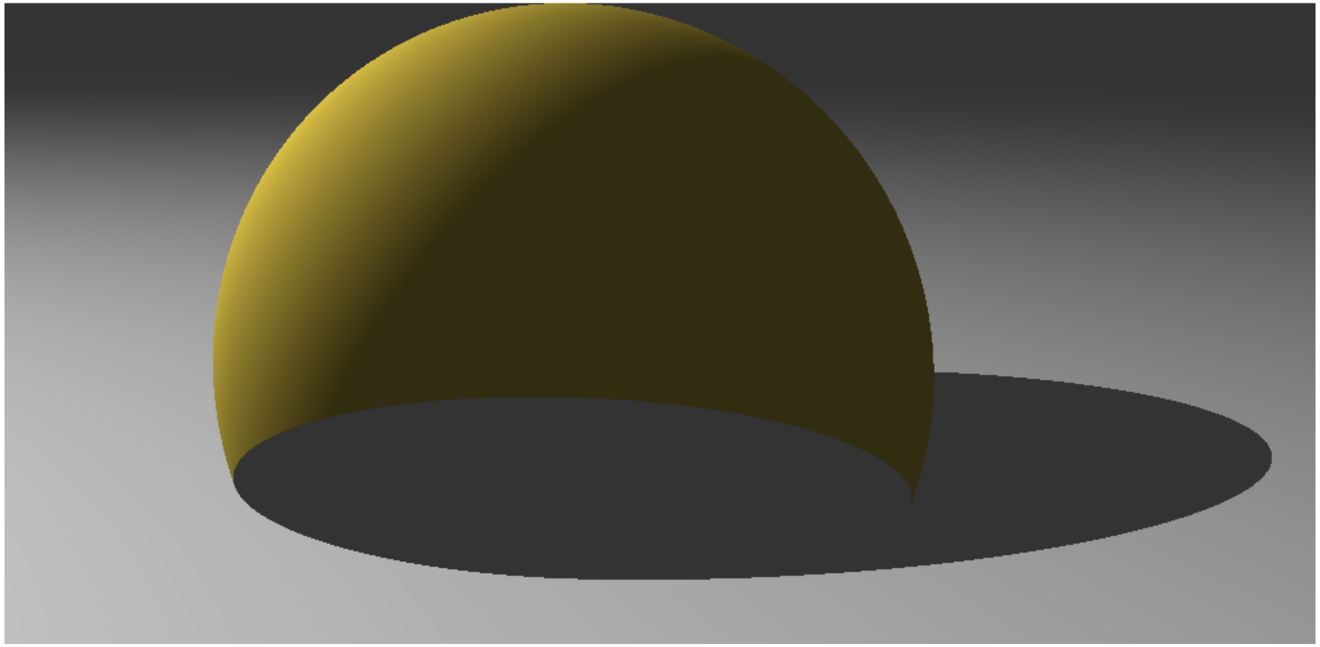
Avantage: Vraie sphère



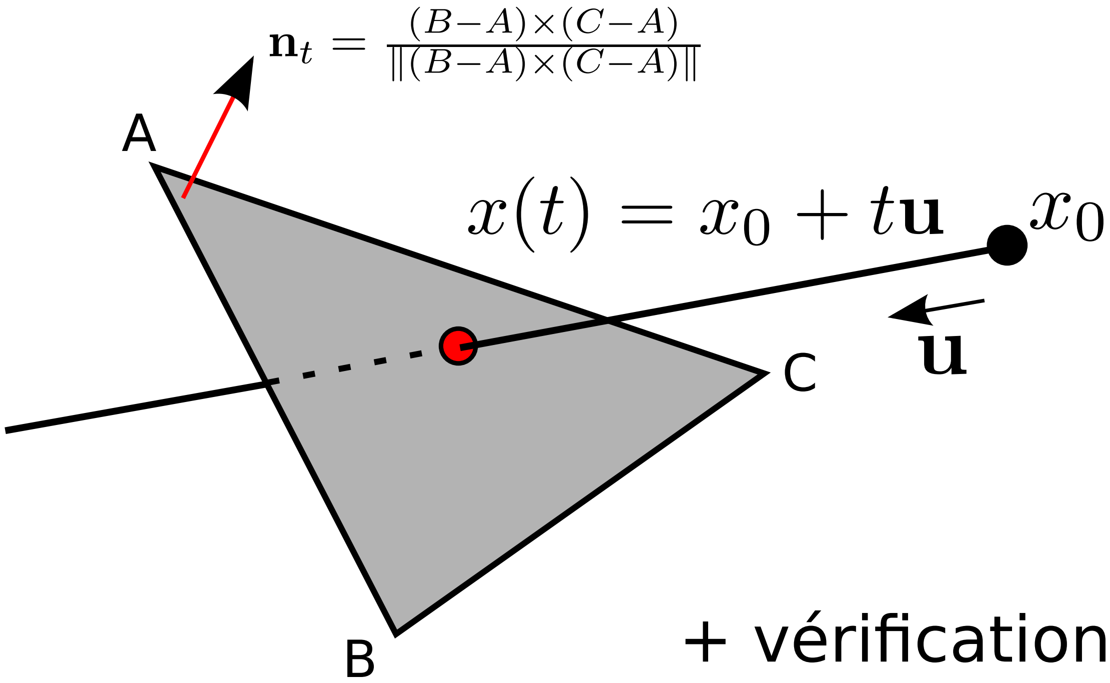
Cas de la sphère

Rem. Inversion d'intersection





Cas du triangle



Similaire plan

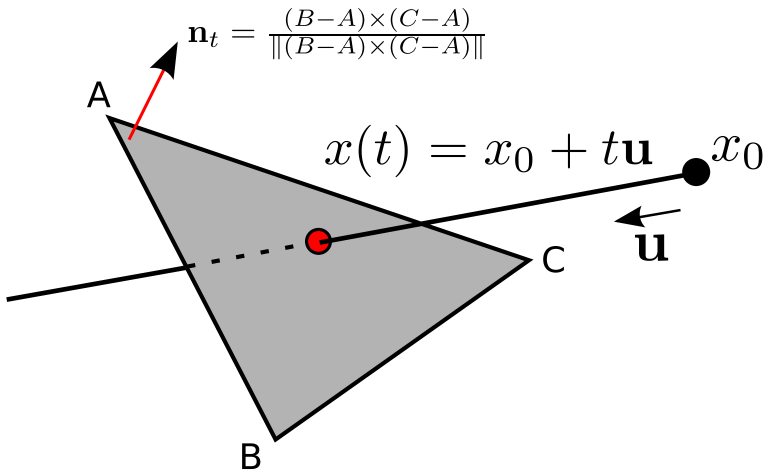
$$t = \frac{<A - x_0, \mathbf{n}_t>}{<\mathbf{u}, \mathbf{n}_t>}$$

+ vérification coordonnées barycentriques

$$x = \alpha A + \beta B + \gamma C$$

$$x \in \mathcal{T} \Rightarrow \left(\begin{array}{l} \alpha + \beta + \gamma = 1 \\ 0 \leq \alpha \leq 1 \\ 0 \leq \beta \leq 1 \\ 0 \leq \gamma \leq 1 \end{array} \right.$$

Coord. barycentriques



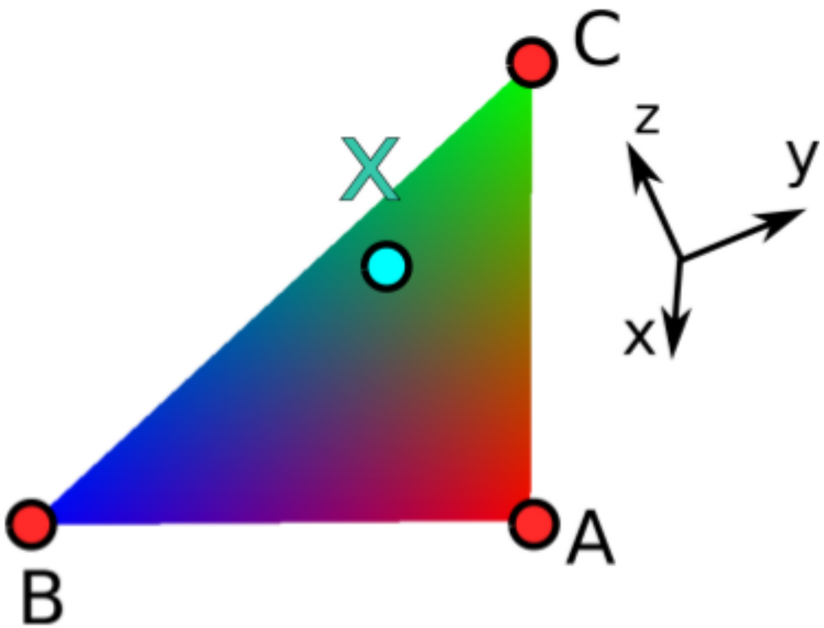
$$x = \alpha A + \beta B + \gamma C$$

$$\begin{cases} \alpha + \beta + \gamma = 1 \\ 0 \leq \alpha \leq 1 \\ 0 \leq \beta \leq 1 \\ 0 \leq \gamma \leq 1 \end{cases}$$

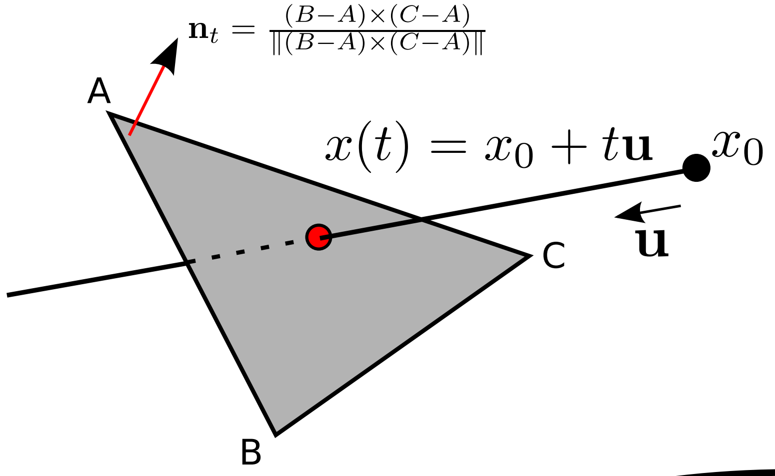
$$\begin{cases} A = aire(\mathbf{x}_B - \mathbf{x}_A, \mathbf{x}_C - \mathbf{x}_A) \\ A_1 = aire(\mathbf{x}_C - \mathbf{x}_B, \mathbf{x} - \mathbf{x}_B) \\ A_2 = aire(\mathbf{x}_A - \mathbf{x}_C, \mathbf{x} - \mathbf{x}_C) \\ A_3 = aire(\mathbf{x}_B - \mathbf{x}_A, \mathbf{x} - \mathbf{x}_A) \end{cases}$$

avec $aire(\mathbf{v}_0, \mathbf{v}_1) = 1/2\|\mathbf{v}_0 \times \mathbf{v}_1\|$

$$\Rightarrow \begin{cases} \alpha = A_1/A \\ \beta = A_2/A \\ \gamma = A_3/A \end{cases}$$



Coord. barycentriques

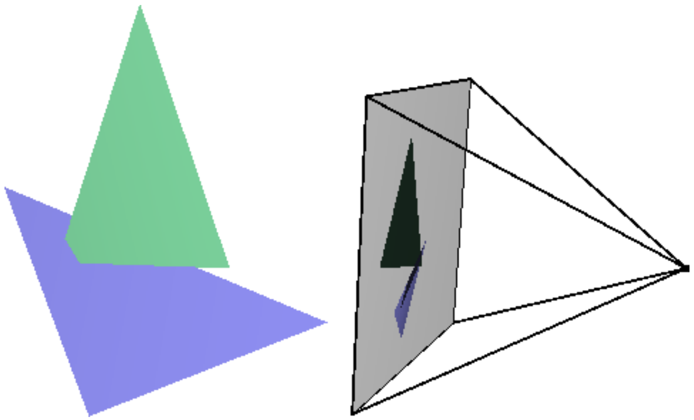


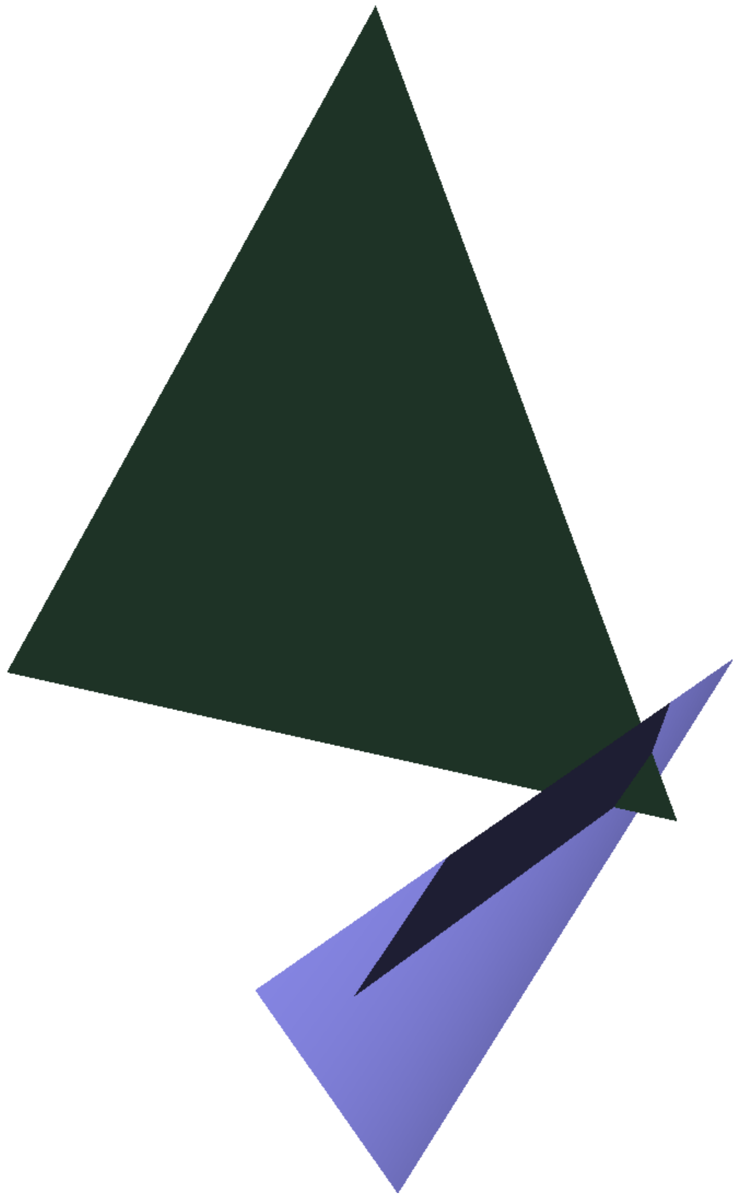
$$t = \frac{<A - x_0, \mathbf{n}_t>}{<\mathbf{u}, \mathbf{n}_t>}$$

$$\left(\begin{array}{l} \alpha + \beta + \gamma = 1 \\ 0 \leq \alpha \leq 1 \\ 0 \leq \beta \leq 1 \\ 0 \leq \gamma \leq 1 \end{array} \right.$$

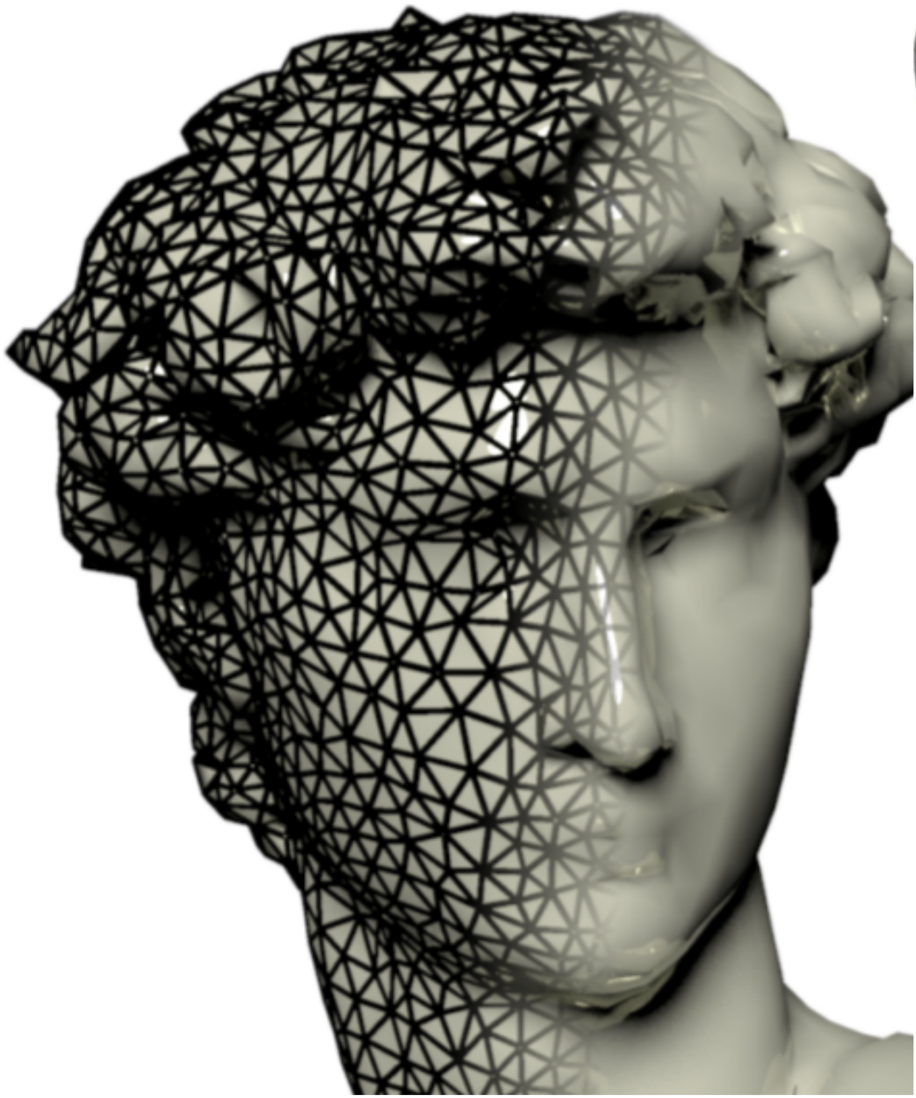
```
v3 x10=internal_x1-internal_x0;
v3 x20=internal_x2-internal_x0;

v3 u10=x10.normalized();
v3 u20=x20.normalized();

v3 n=u10.cross(u20).normalized();

const v3& u=seg.u();
double proj=u.dot(n);

double epsilon=1e-8;
if(std::fabs(proj)<epsilon)
    return inter;

double t=(internal_x0-seg.x0()).dot(n)/proj;
v3 xi=seg(t);

double area_0=(internal_x2-internal_x1).cross(xi-internal_x1).norm()/2.0;
double area_1=(internal_x0-internal_x2).cross(xi-internal_x2).norm()/2.0;
double area_2=(internal_x1-internal_x0).cross(xi-internal_x0).norm()/2.0;
double area  =x10.cross(x20).norm()/2.0;

double a=area_0/area;
double b=area_1/area;
double c=area_2/area;


if(a>=0 && b>=0 && c>=0 && a<=1 && b<=1 && c<=1)
    if(std::fabs(a+b+c-1.0)<epsilon)
        inter.push_back(intersection_data(xi,n,t));

return inter;
```

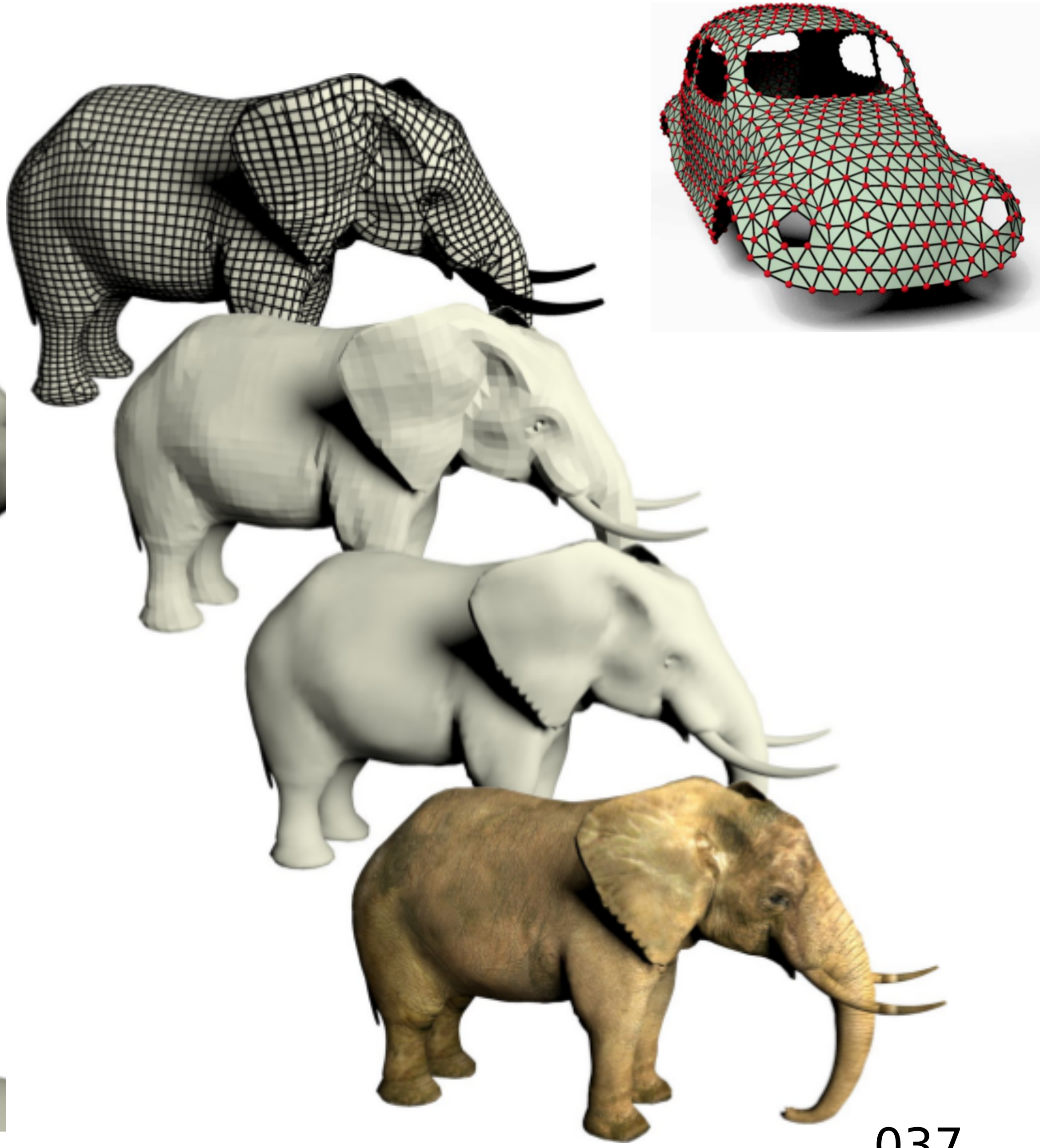
Cas du triangle





Triangle => Maillage





Surfaces implicites

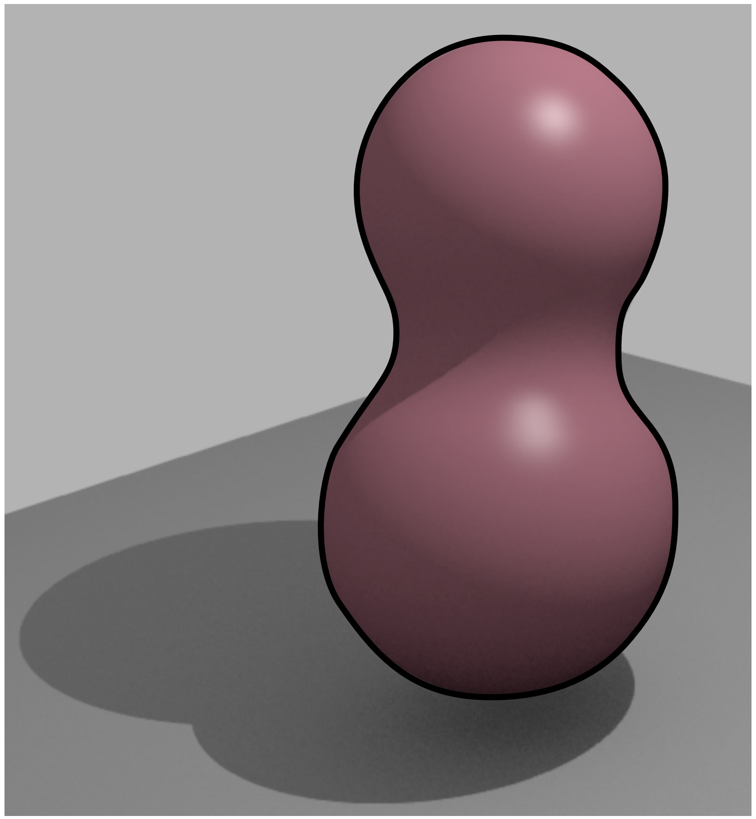
$$S = \{(x,y,z) \in \mathbb{R}^3 | F(x,y,z) = 0\}$$

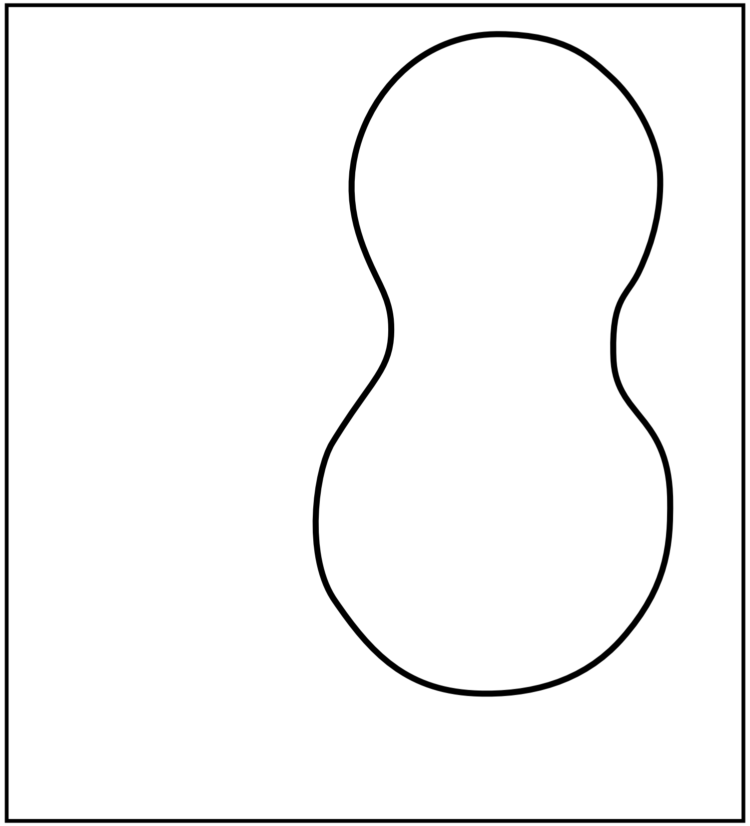
De manière générale, on ne connait pas analytiquement $S \cap \mathcal{D}$

On recherche une approximation

Surfaces implicites

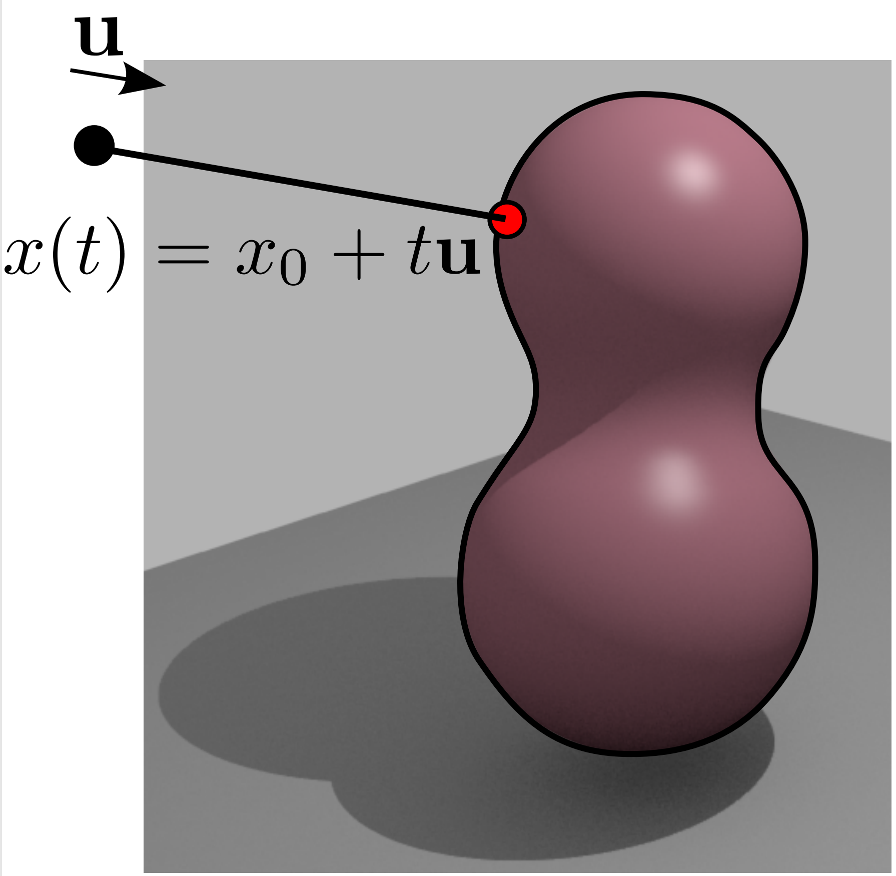
$$S = \{(x,y,z) \in \mathbb{R}^3 | F(x,y,z) = 0\}$$

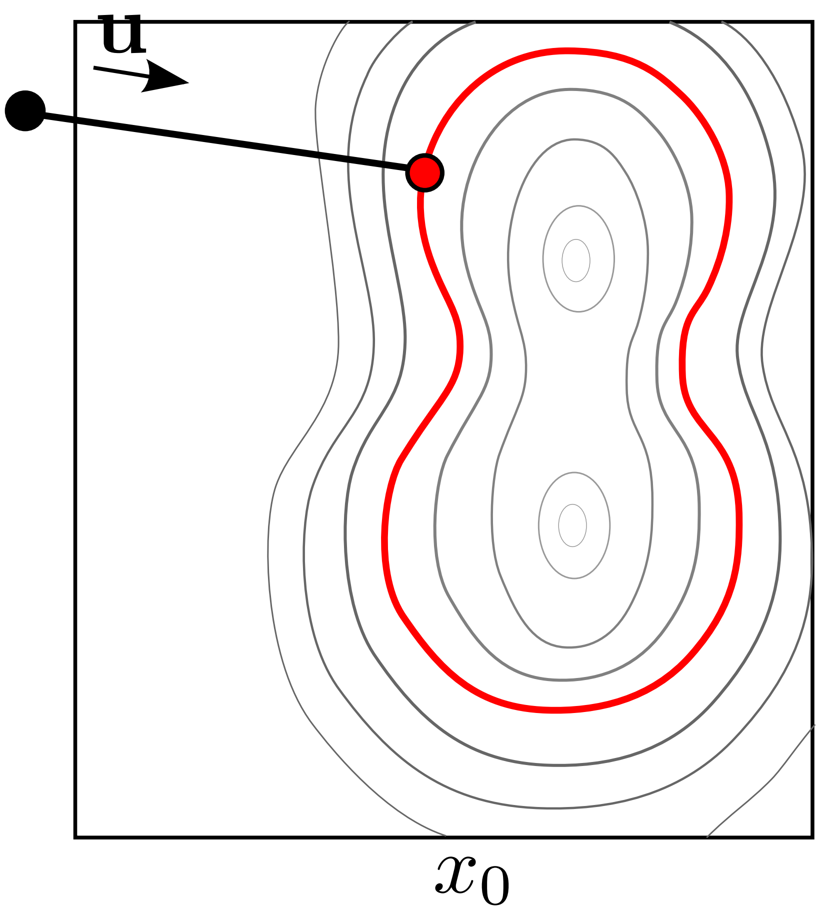




Surfaces implicites

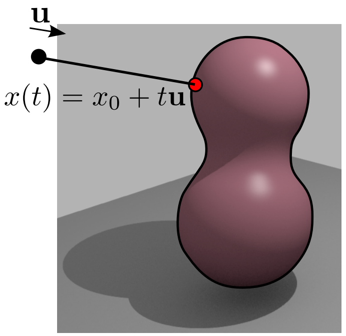
$$S = \{(x, y, z) \in \mathbb{R}^3 | F(x, y, z) = 0\}$$

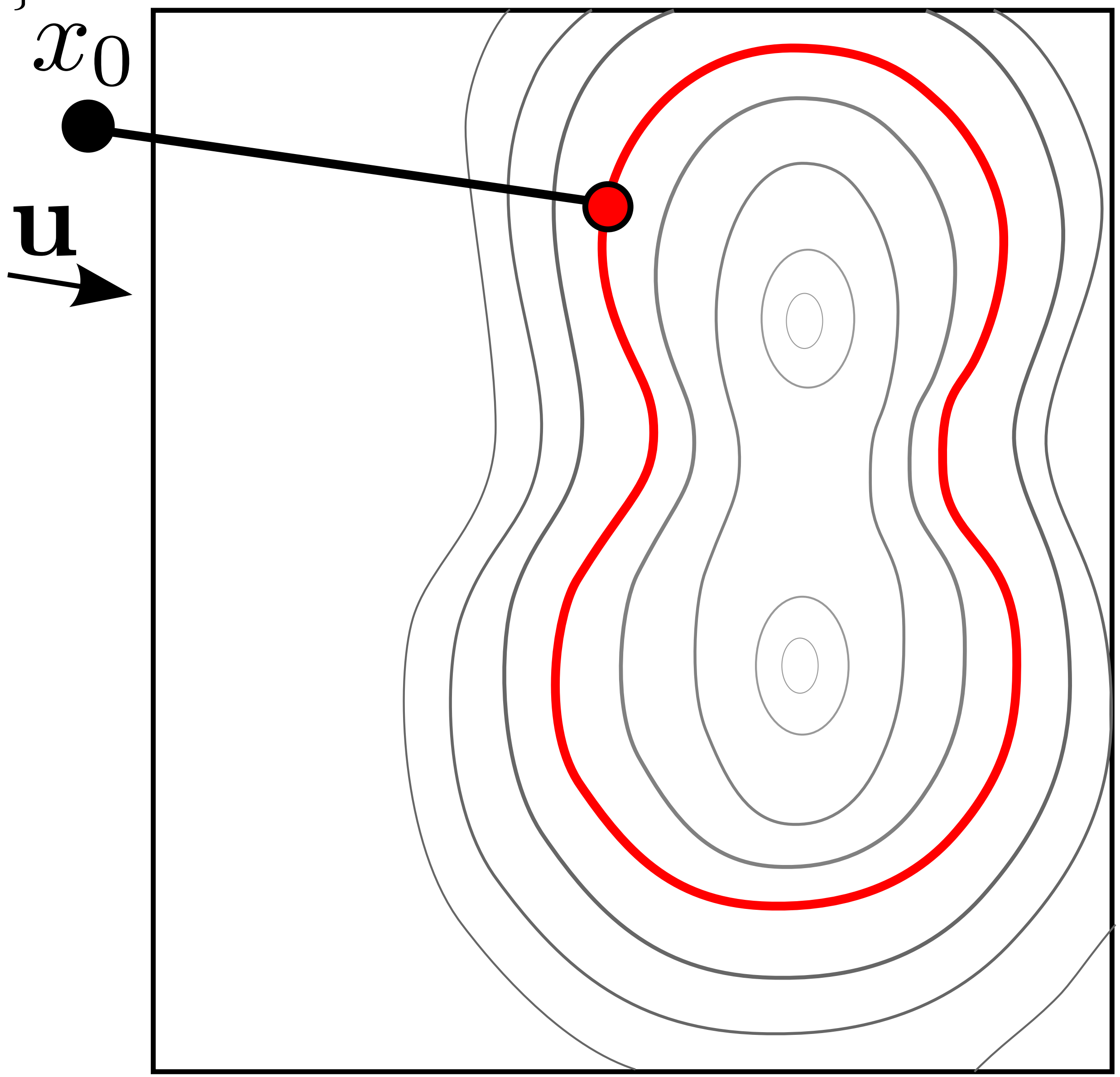




Surfaces implicites

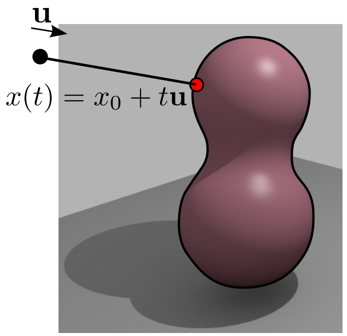
$$S = \{(x, y, z) \in \mathbb{R}^3 | F(x, y, z) = 0\}$$





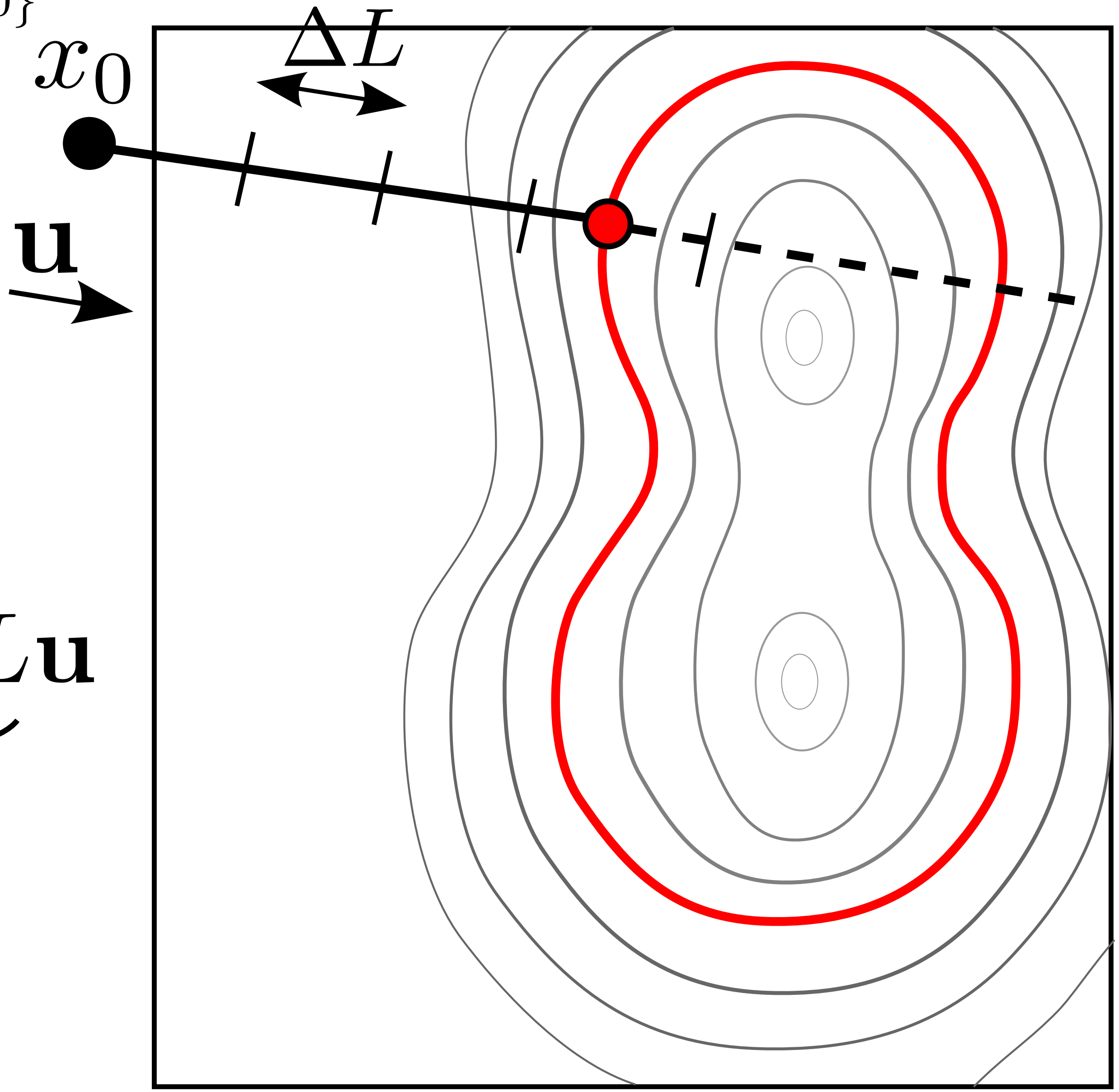
Surfaces implicites

$$S = \{(x,y,z) \in \mathbb{R}^3 | F(x,y,z) = 0\}$$



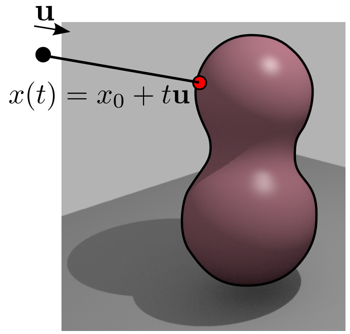
Solution 1:

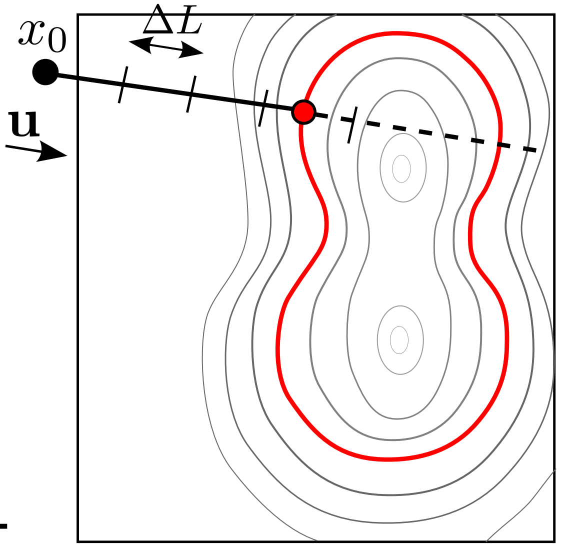
$$x^{i+1} = x^i + \underbrace{\Delta L}_{?}\mathbf{u}$$



Surfaces implicites

$S = \{(x,y,z) \in \mathbb{R}^3 | F(x,y,z) = 0\}$

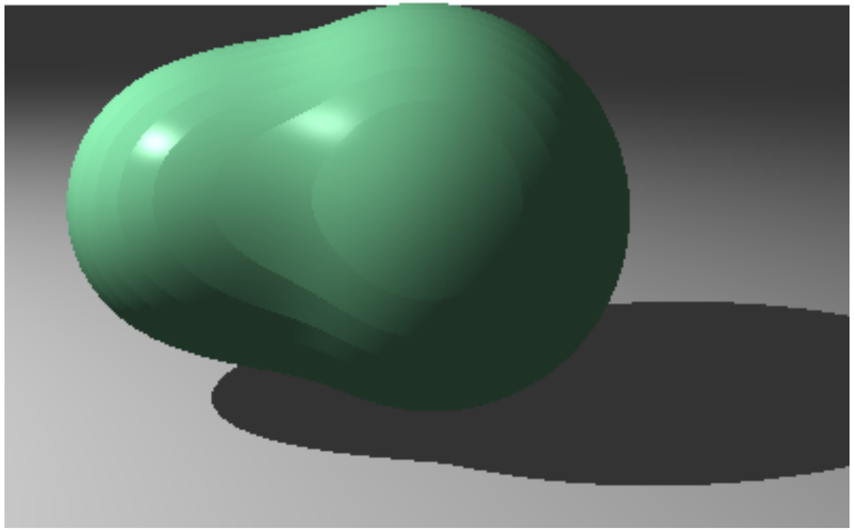




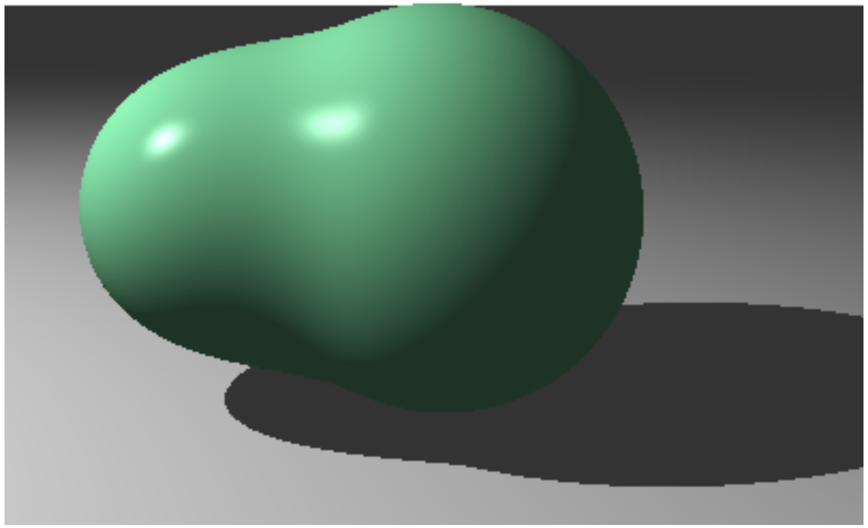
Solution 1:

$$x^{i+1} = x^i + \Delta L \mathbf{u}$$

0.2

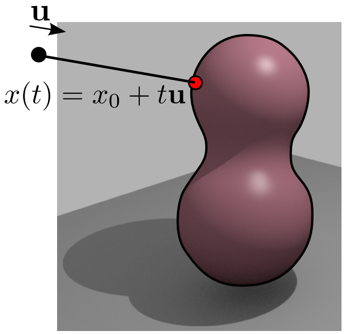


0.001



Surfaces implicites

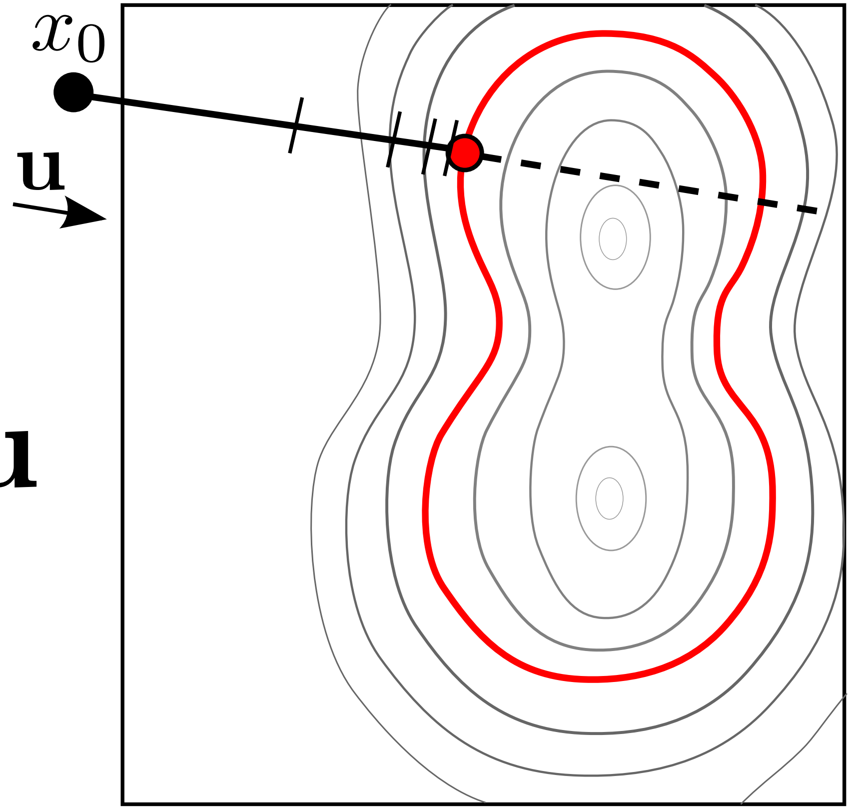
$S = \{(x,y,z) \in \mathbb{R}^3 | F(x,y,z) = 0\}$



F: dérivable

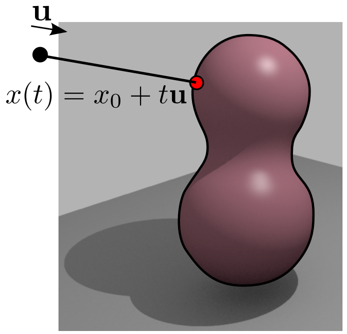
Solution 2:

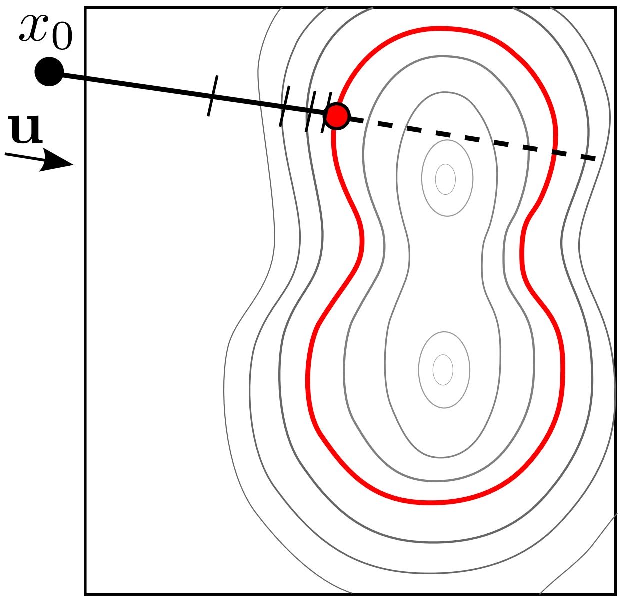
$$x^{i+1} = x^i + \frac{|F(x^i)|}{\|\nabla F\|_{\max}}\mathbf{u}$$



Surfaces implicites

$$S = \{(x,y,z) \in \mathbb{R}^3 | F(x,y,z) = 0\}$$





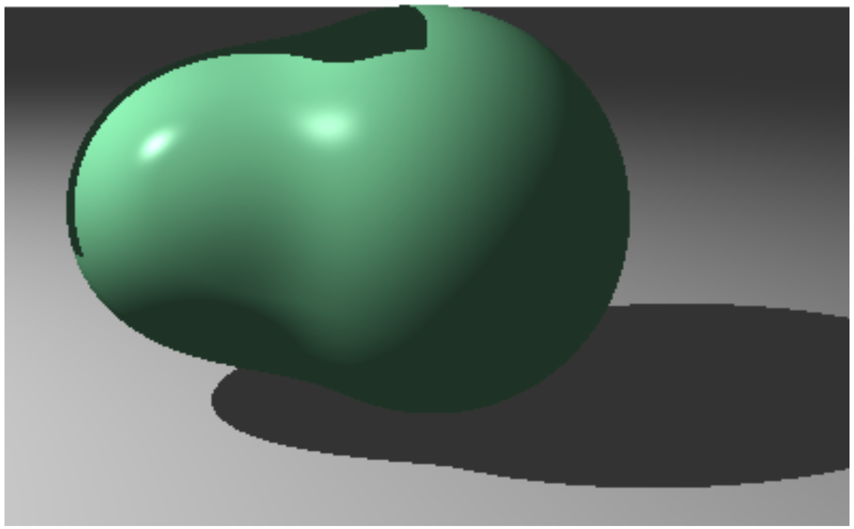
Solution 2:

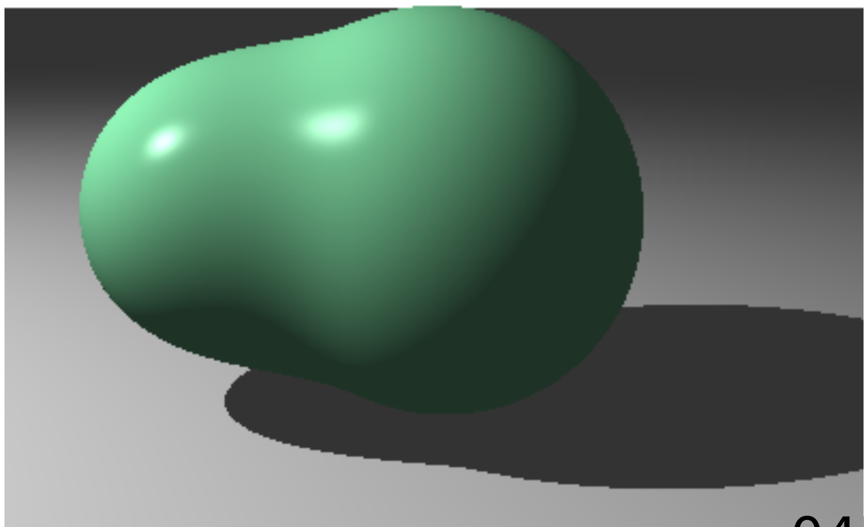
$$x^{i+1} = x^i + \frac{|F(x^i)|}{\|\nabla F\|_{\max}}\mathbf{u}$$

estimé

1

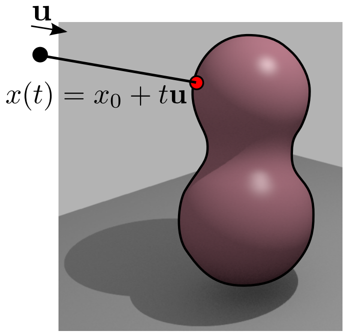
4





Surfaces implicites

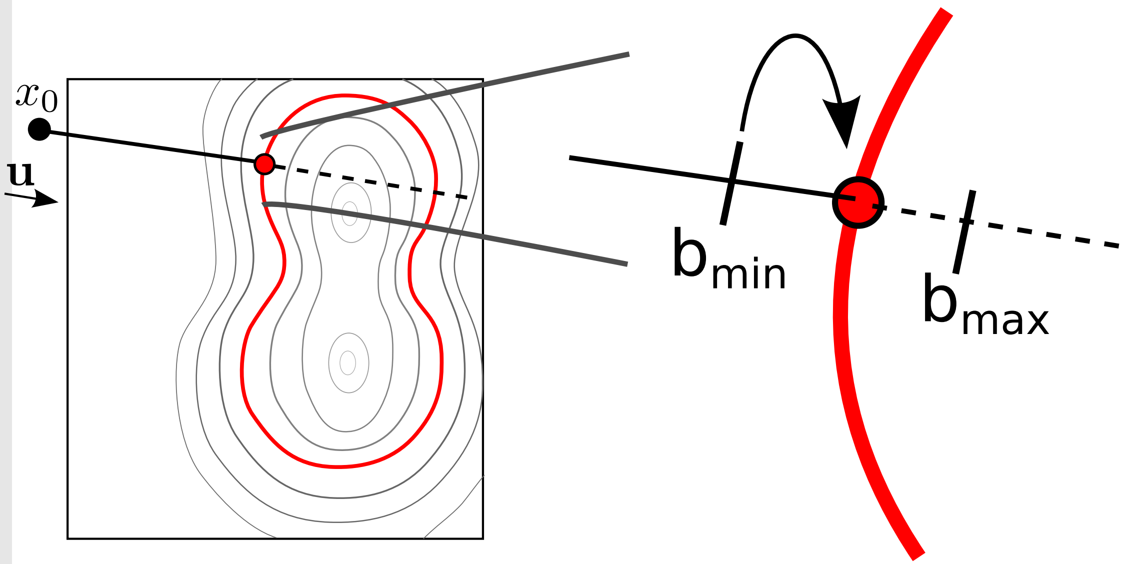
$$S = \{(x,y,z) \in \mathbb{R}^3 | F(x,y,z) = 0\}$$



Solution 2:

$$x^{i+1} = x^i + \frac{|F(x^i)|}{\|\nabla F\|_{\max}} \mathbf{u}$$

Acceleration convergence:



1: Dichotomie

2: Newton
à la fin

$$\boxed{x^{i+1} = x^i - \frac{F(x^i)}{<\nabla F(x^i), \mathbf{u}>}}$$

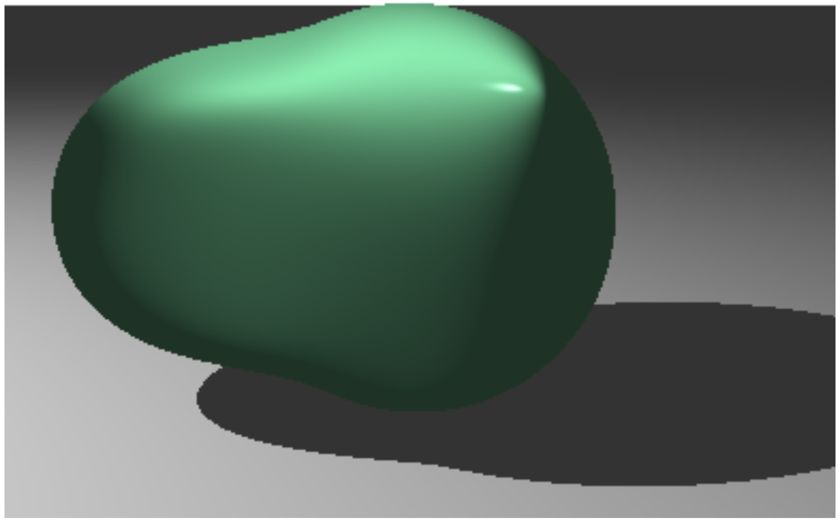
quadratique

Surfaces implicites

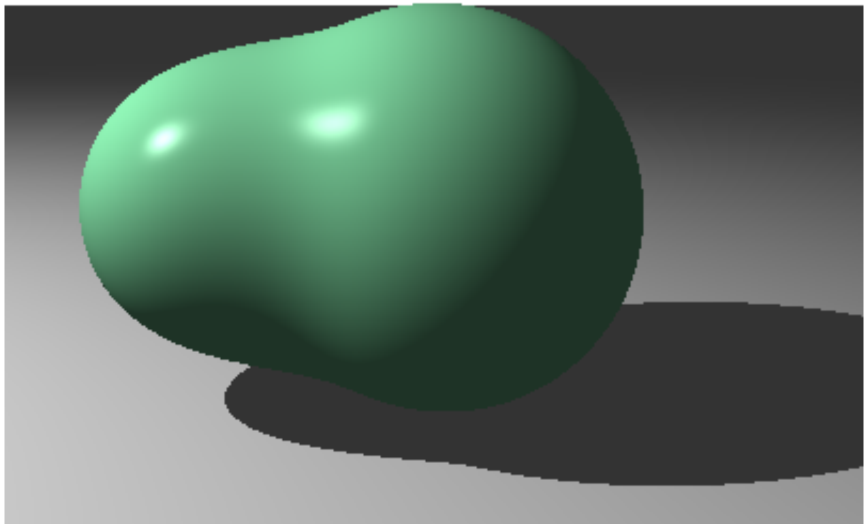
$S = \{(x,y,z) \in \mathbb{R}^3 | F(x,y,z) = 0\}$

Importance du gradient:

$$\frac{F(x+h)-F(x)}{h}$$



∇F analytique

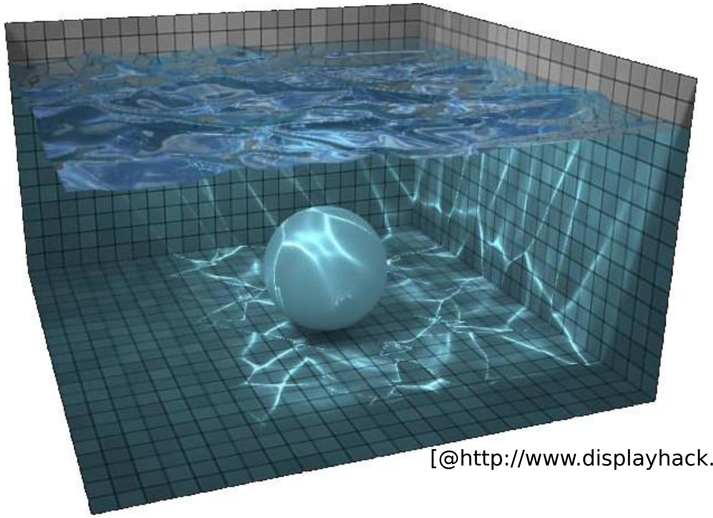


Attention à la convergence

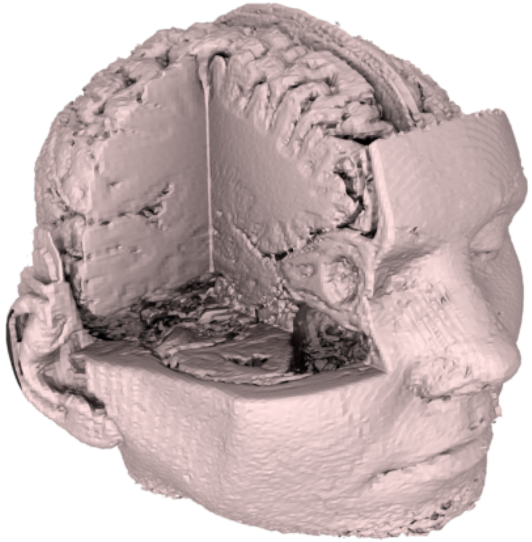
Surfaces implicites

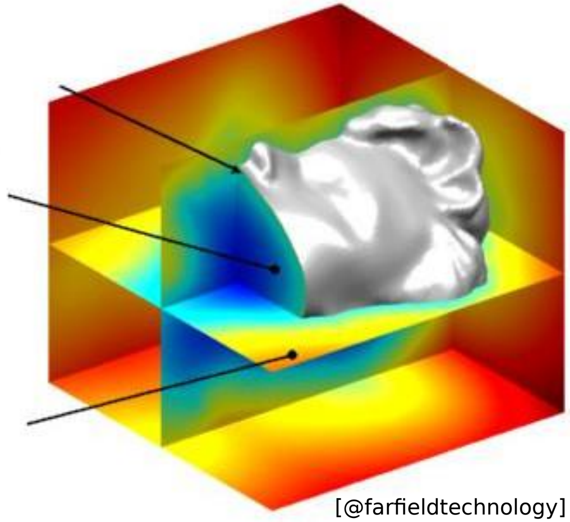
$$S = \{(x,y,z) \in \mathbb{R}^3 | F(x,y,z) = 0\}$$

Données médicales



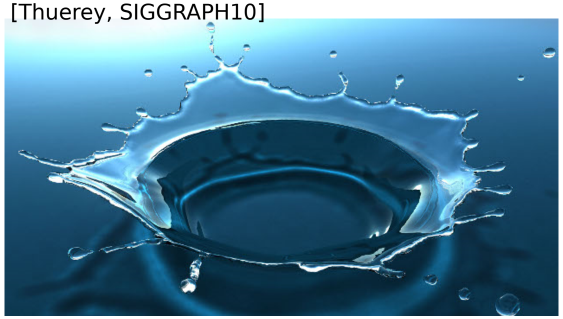
[@http://www.displayhack.org/]



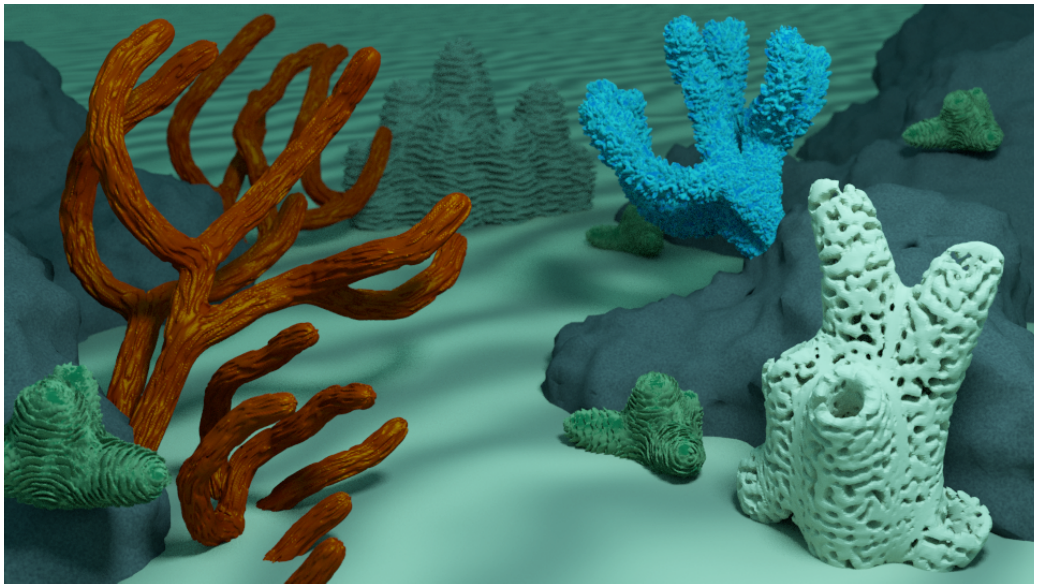


[@farfieldtechnology]

[Thuerey, SIGGRAPH10]



Simulation de fluides



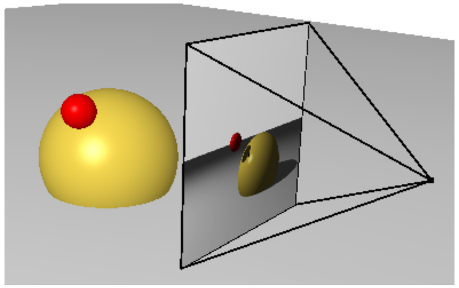
[Zanni, EG12]

Modélisation

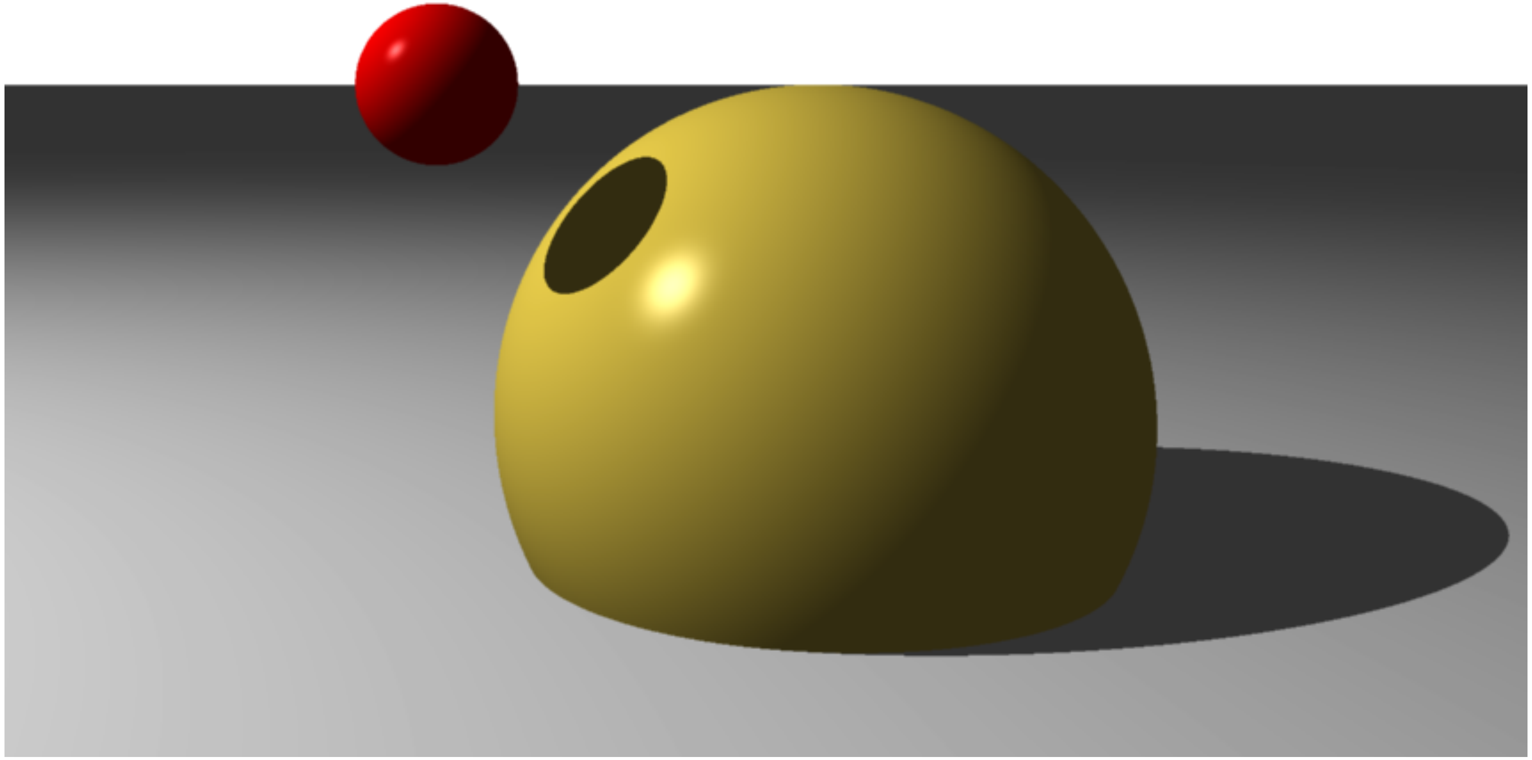
Illumination

Effets visuels

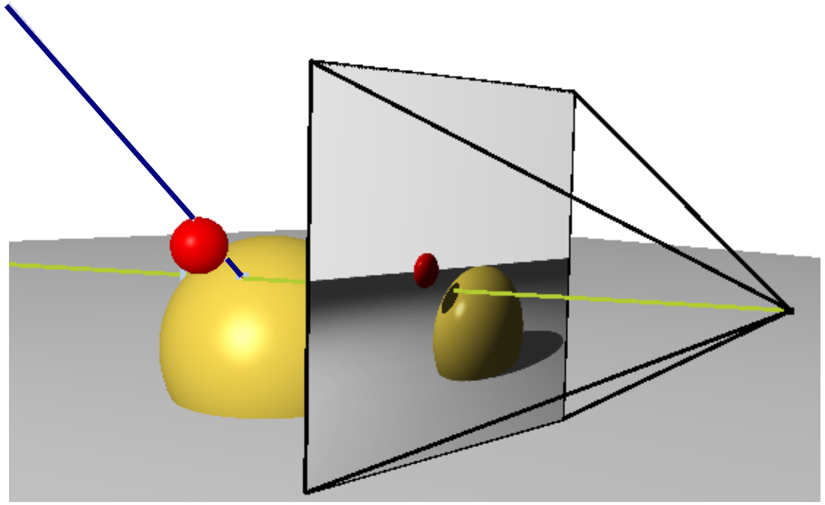
Ombres

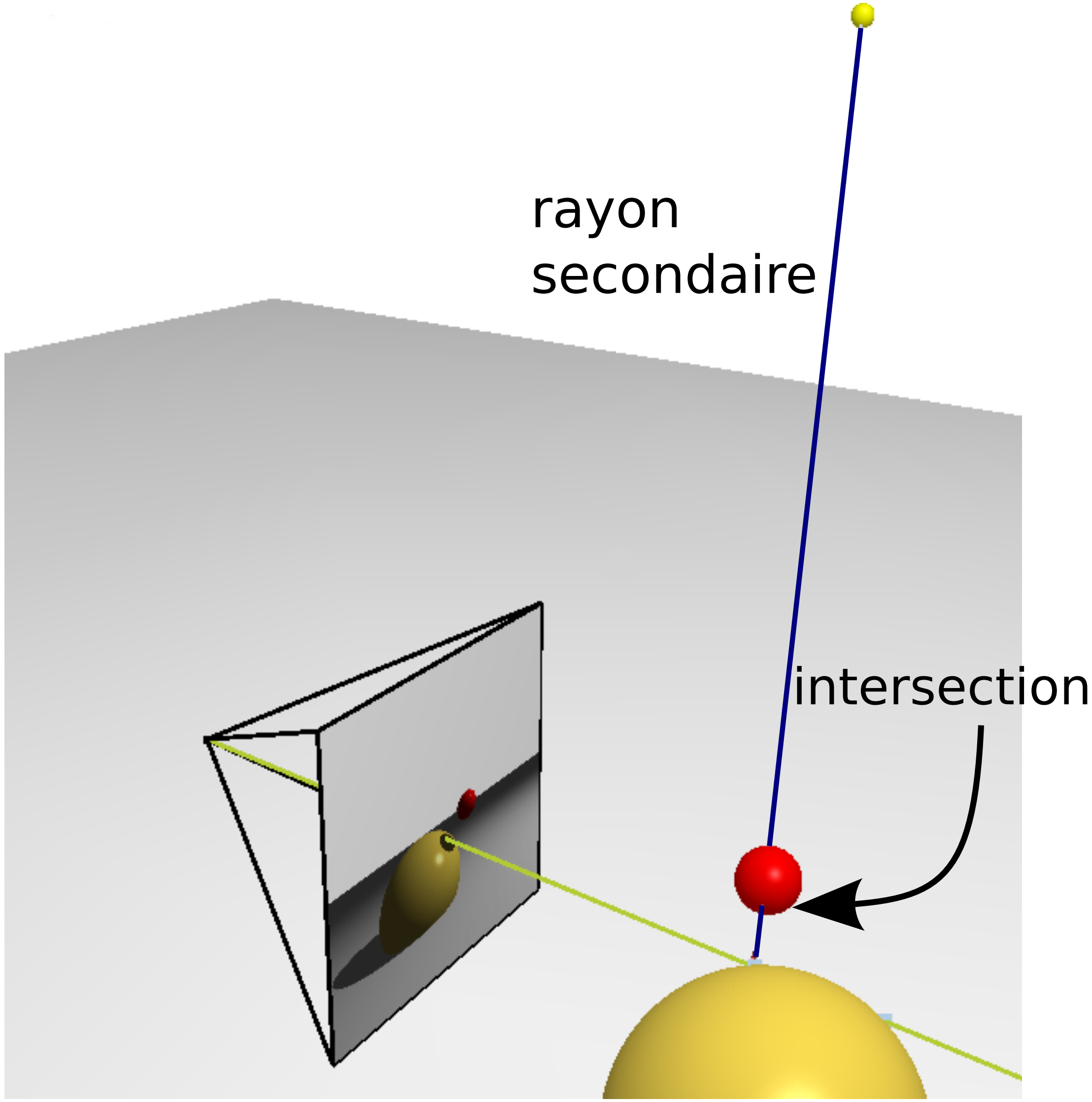


Obtenues naturellement

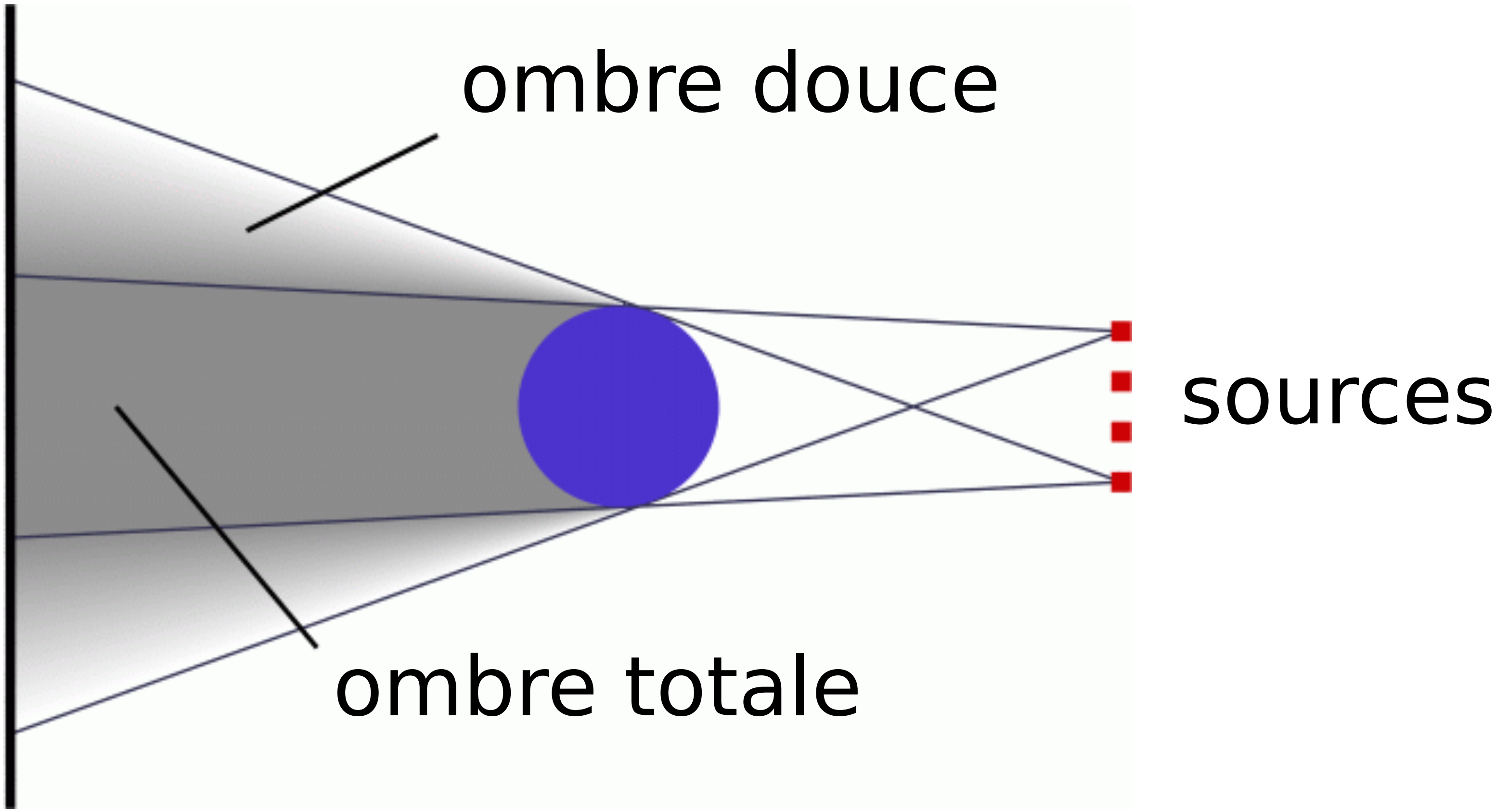


Ombres

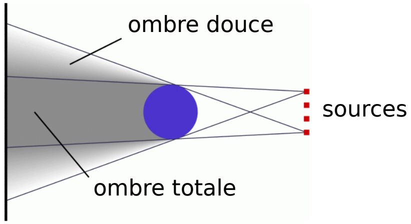


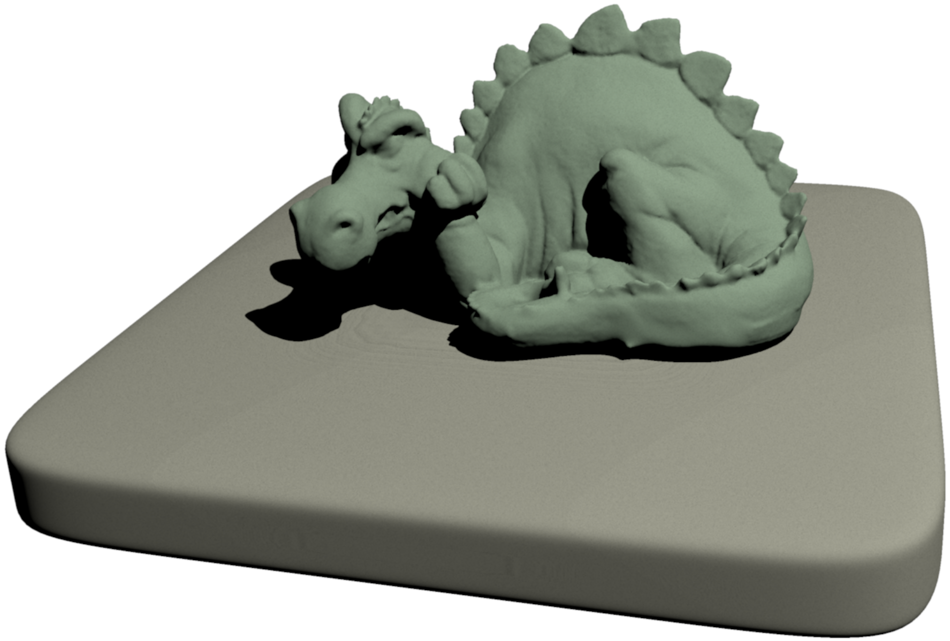


Ombres douces

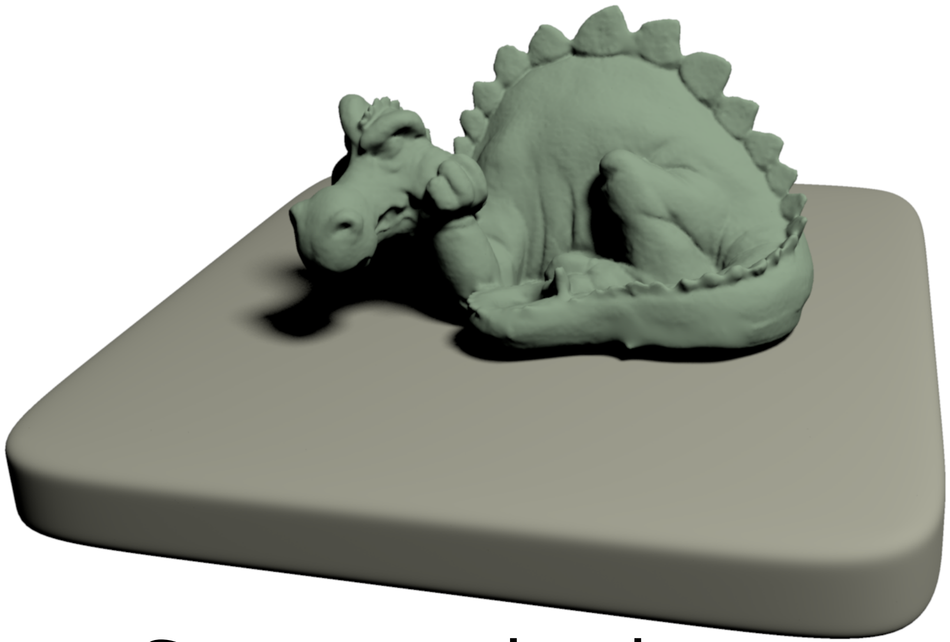


Ombres douces



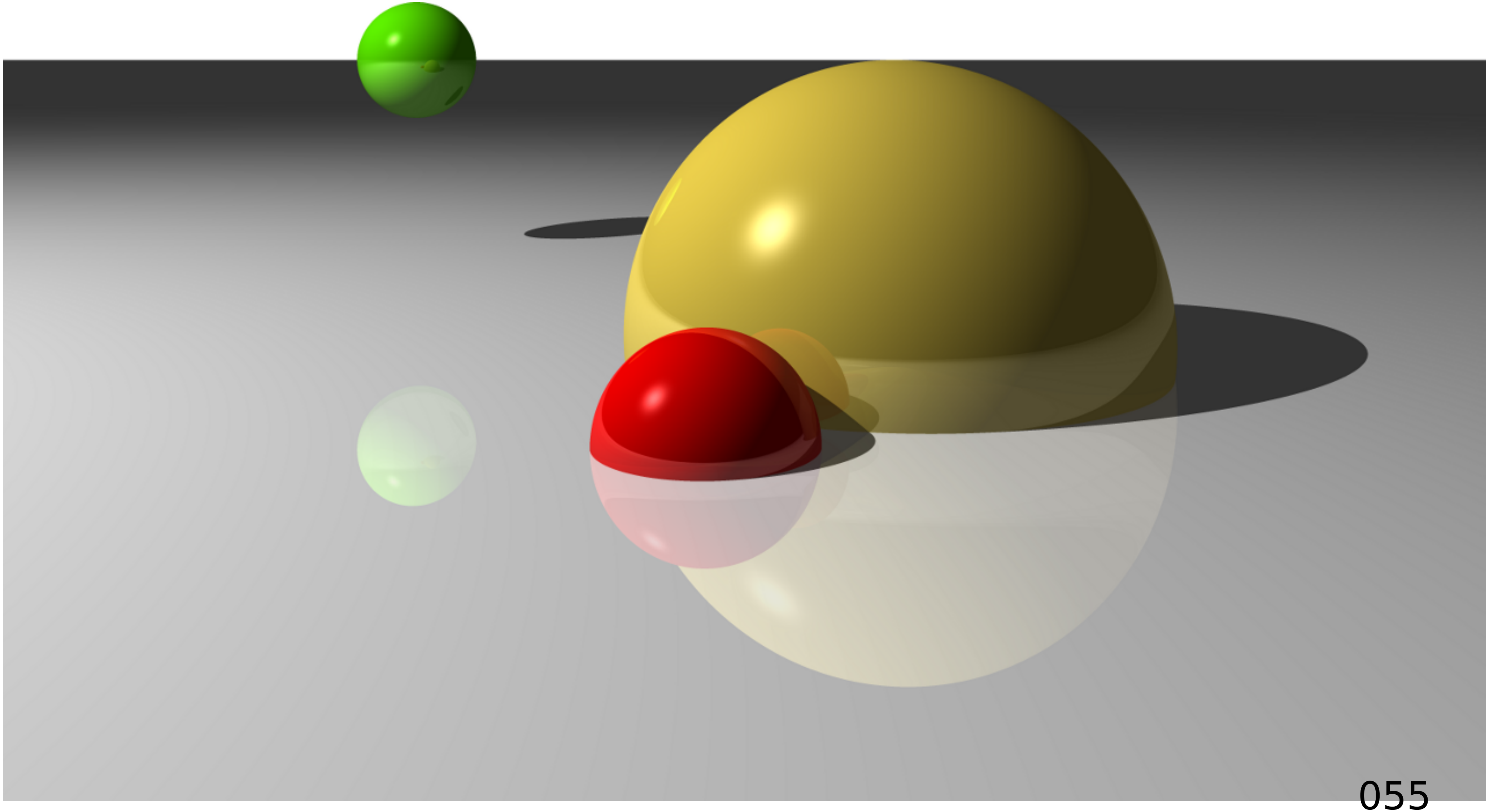


Source ponctuelle

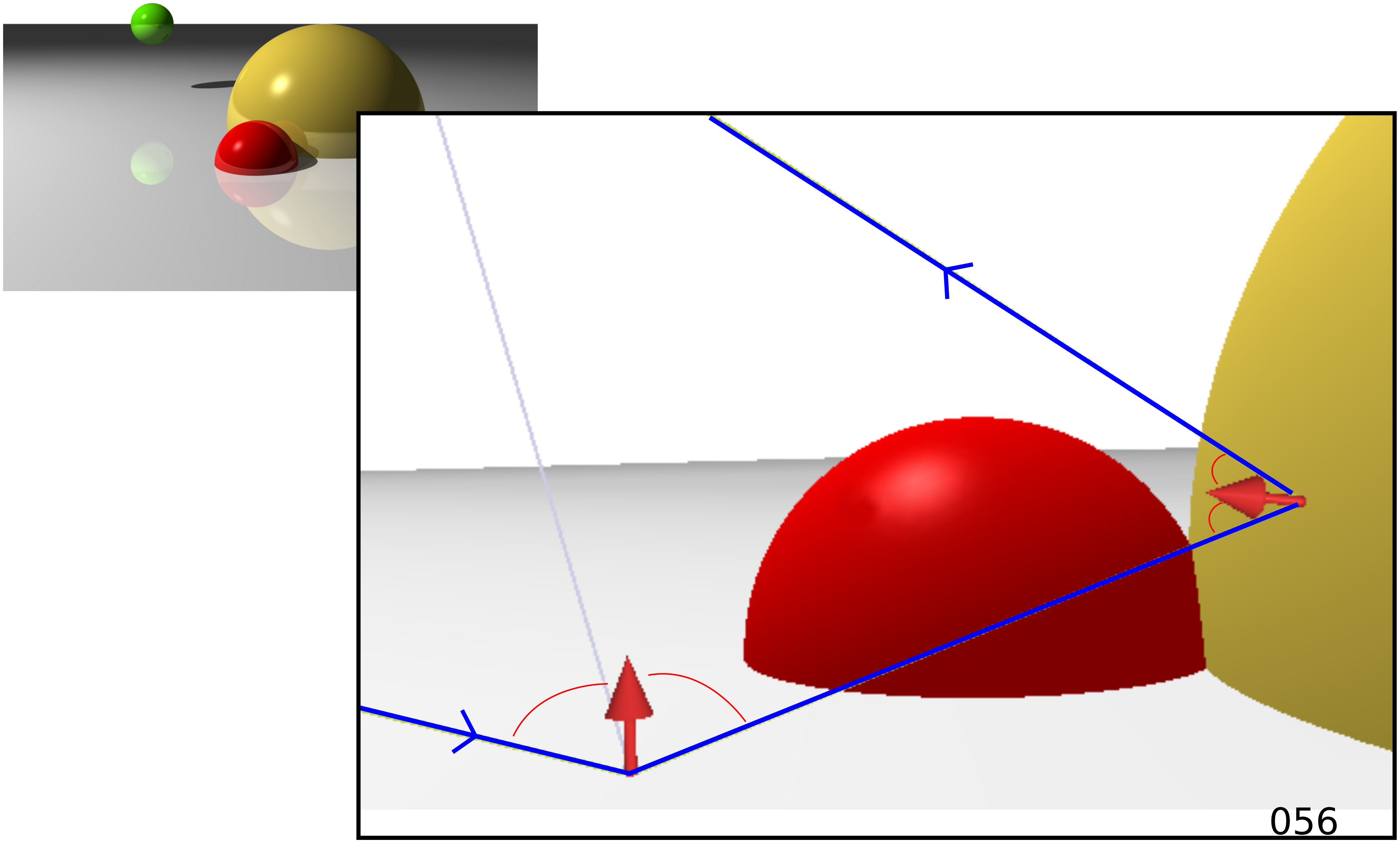


Source spherique

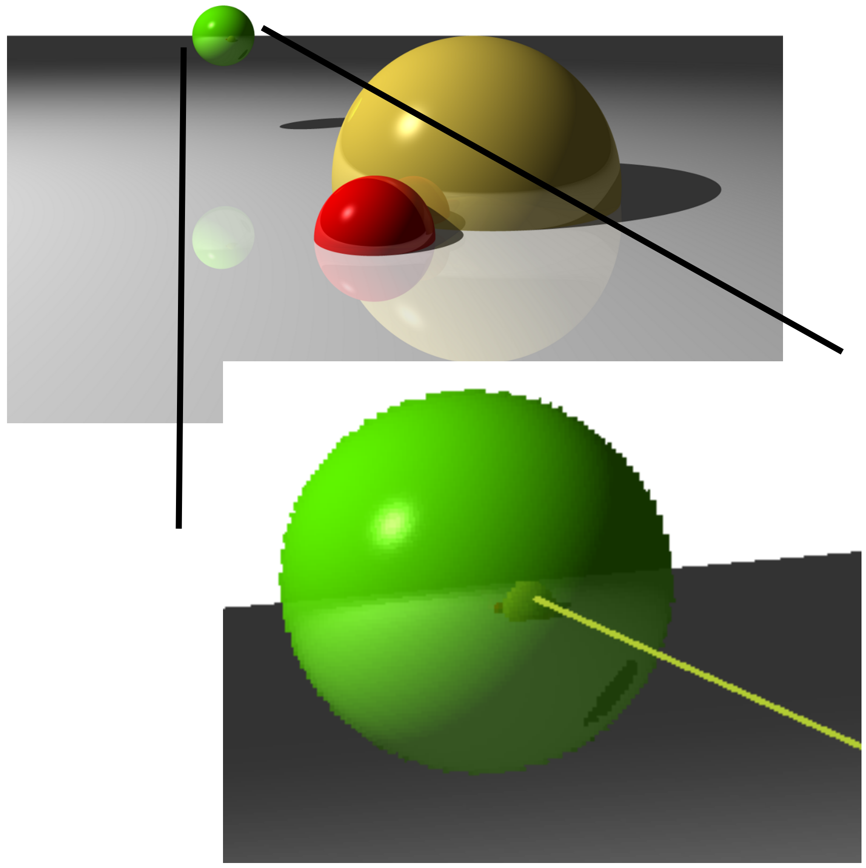
Reflexions

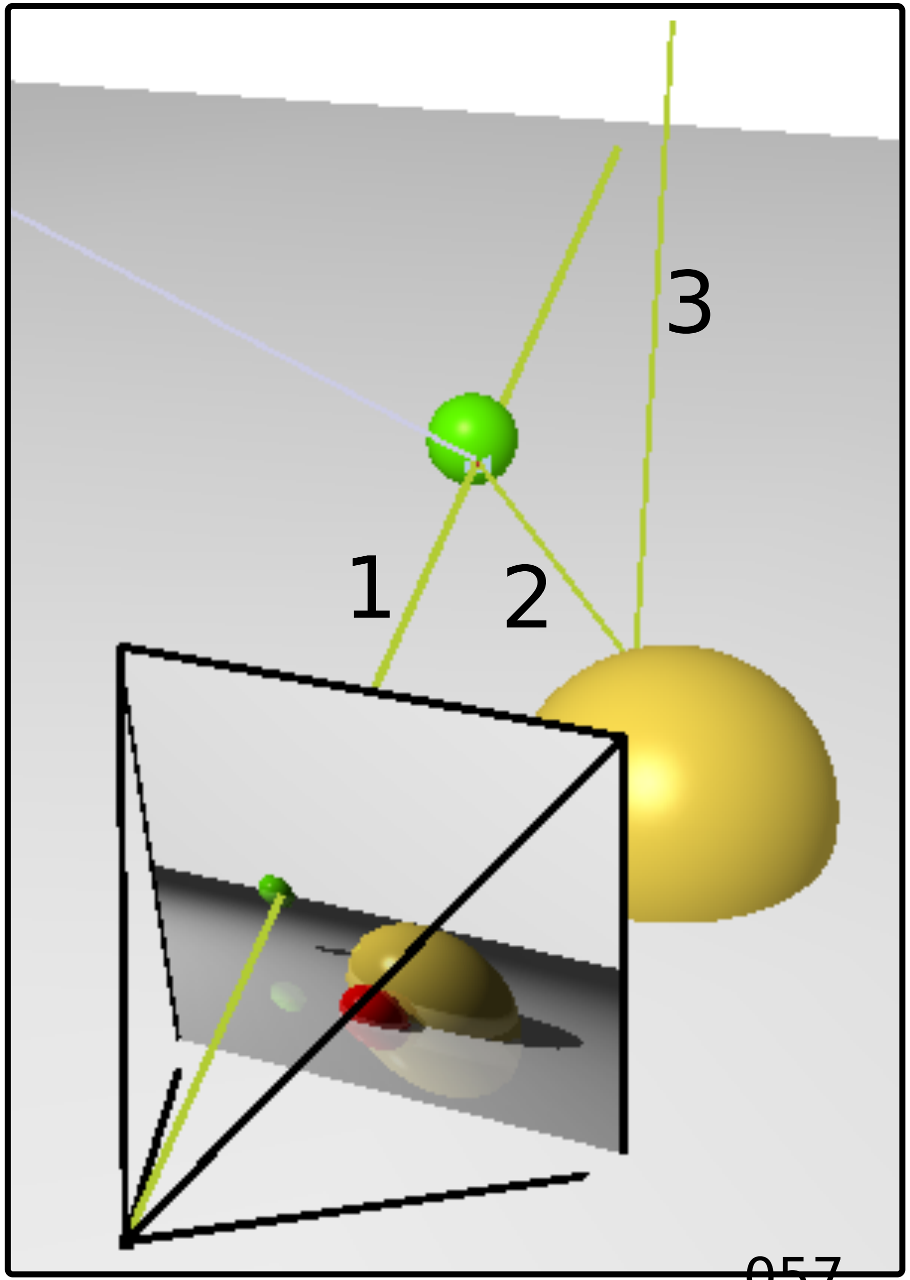


Reflexions



Reflexions

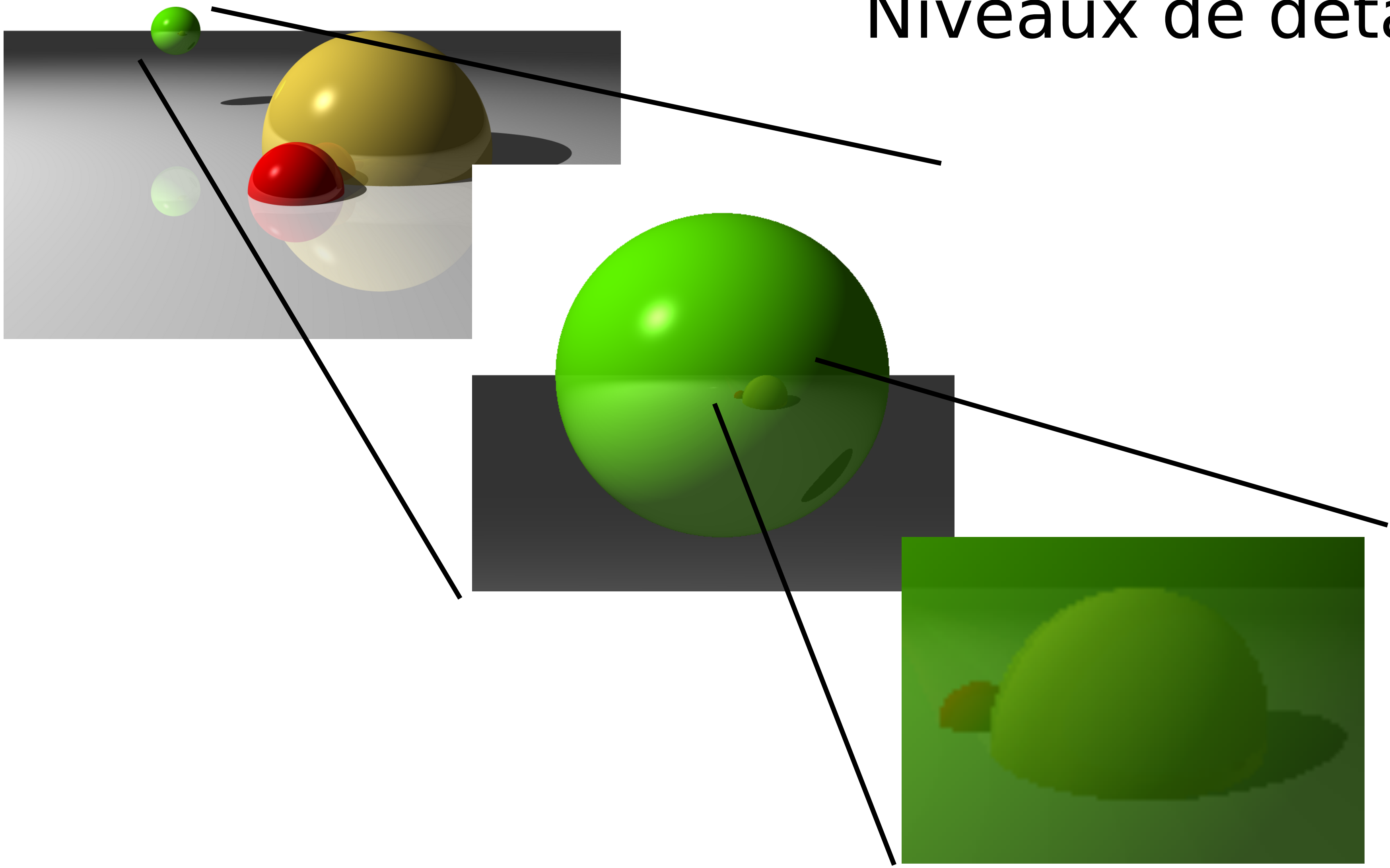




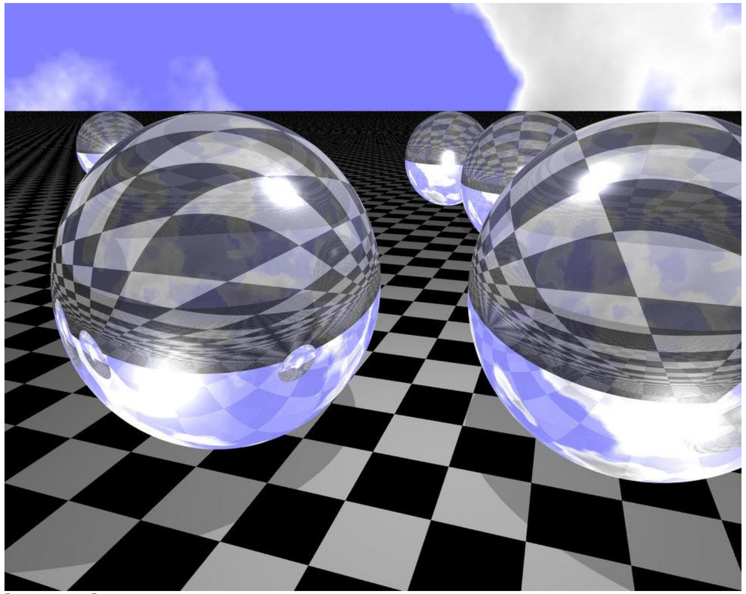
Plusieurs niveaux de reflexions:

Reflexions

Niveaux de détails

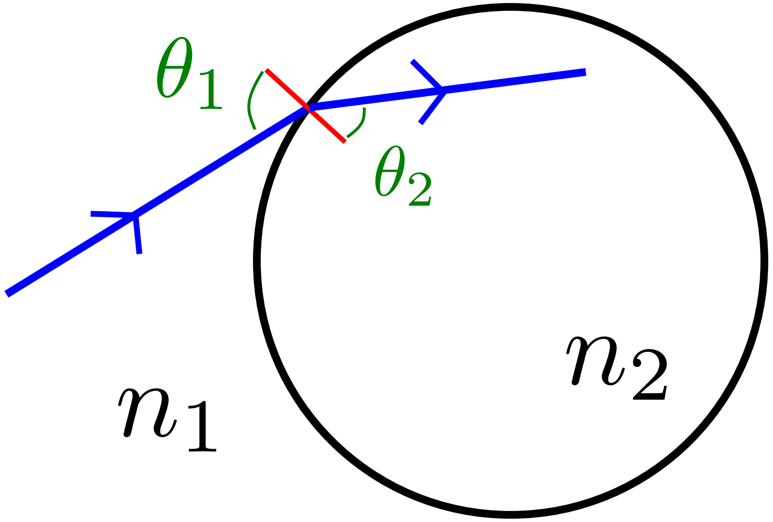


Refraction



[PovRay]

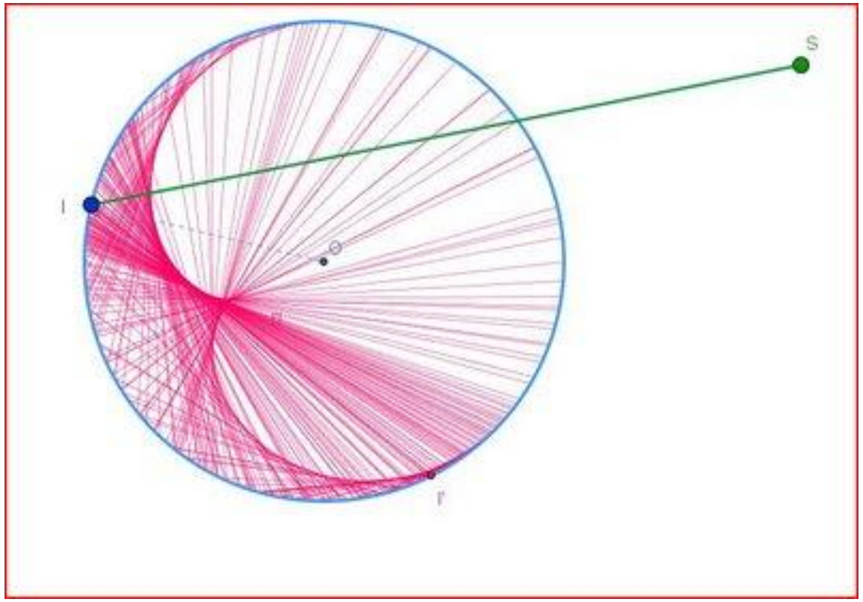
$$n_1 \sin(\theta_1) = n_2 \sin(\theta_2)$$



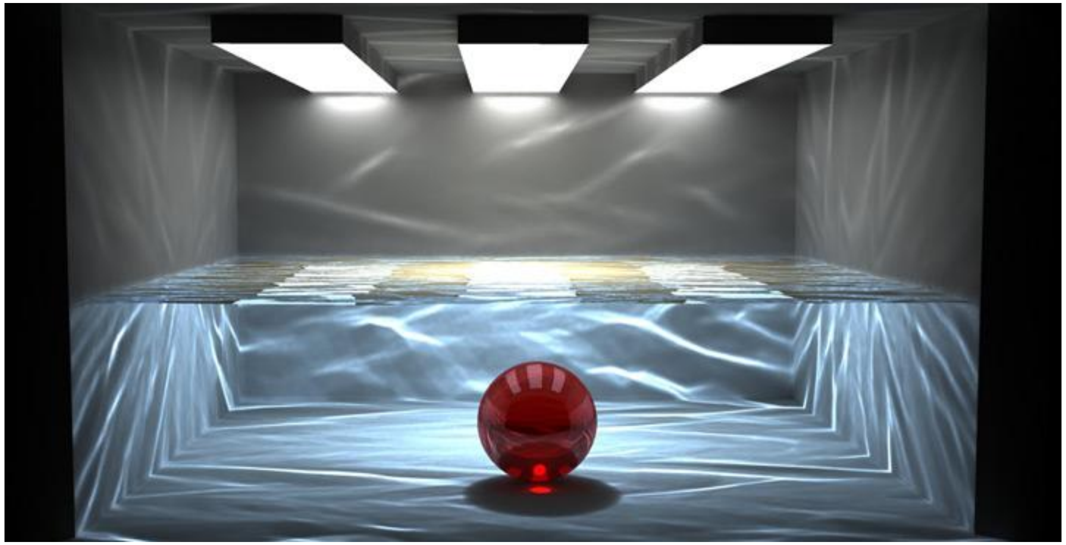
Caustics

Trajet *privilégié* des rayons réfléchis/refractés

[http://abcmathsblog.blogspot.fr/]



Caustics

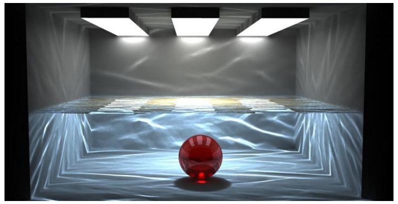


[CG Arena]

[deviantart]

N'est pas obtenu par le principe de base

Caustics

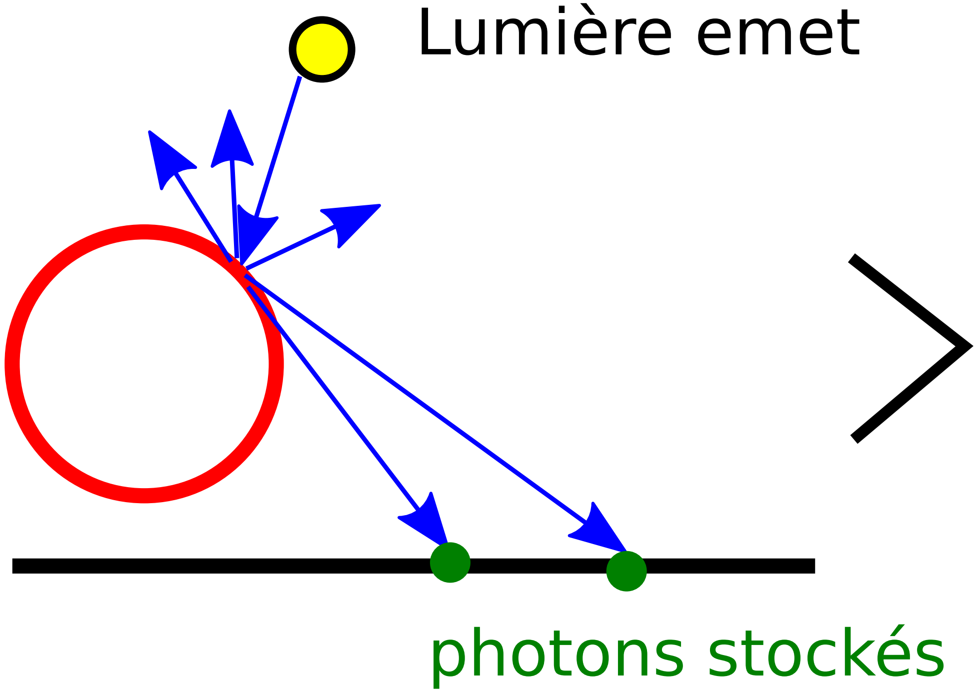


[CG Arena]

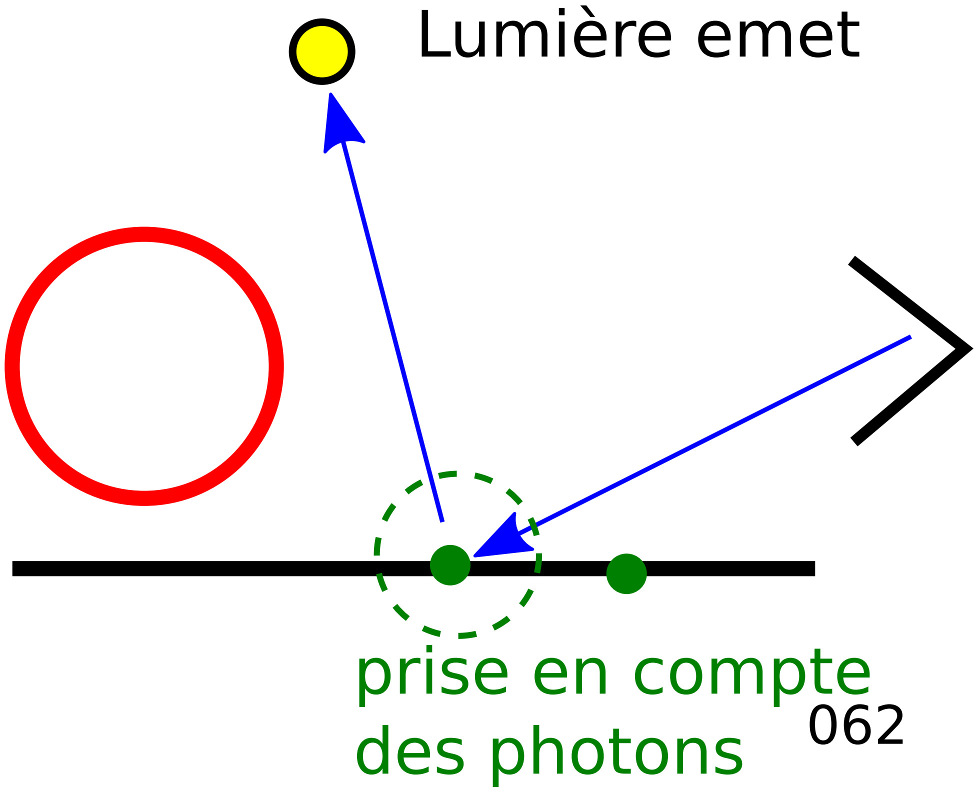
[deviantart]

Utilisation d'une *photon map*

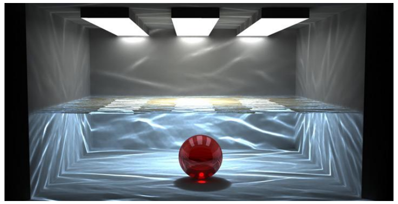
Etape 1:
Lancé de *photons*



Etape 2:
Lancé de rayons



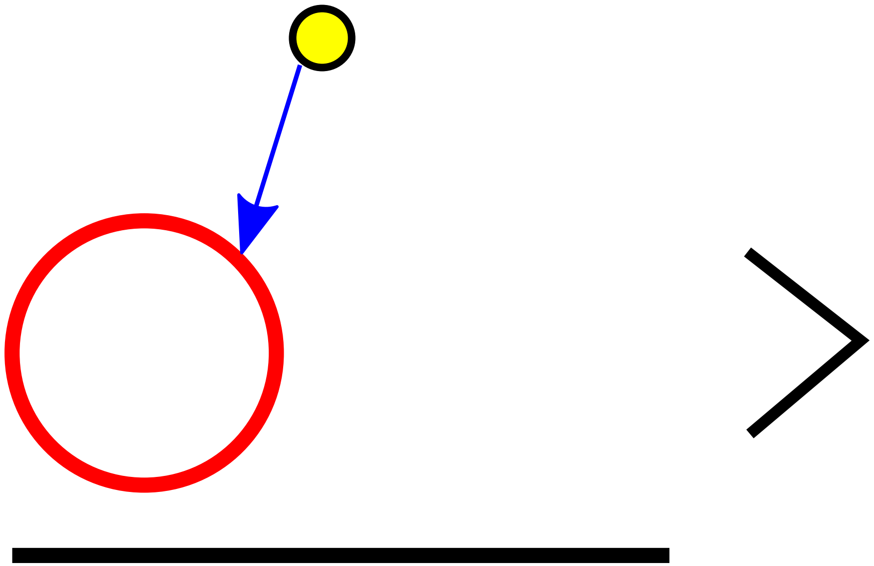
Caustics



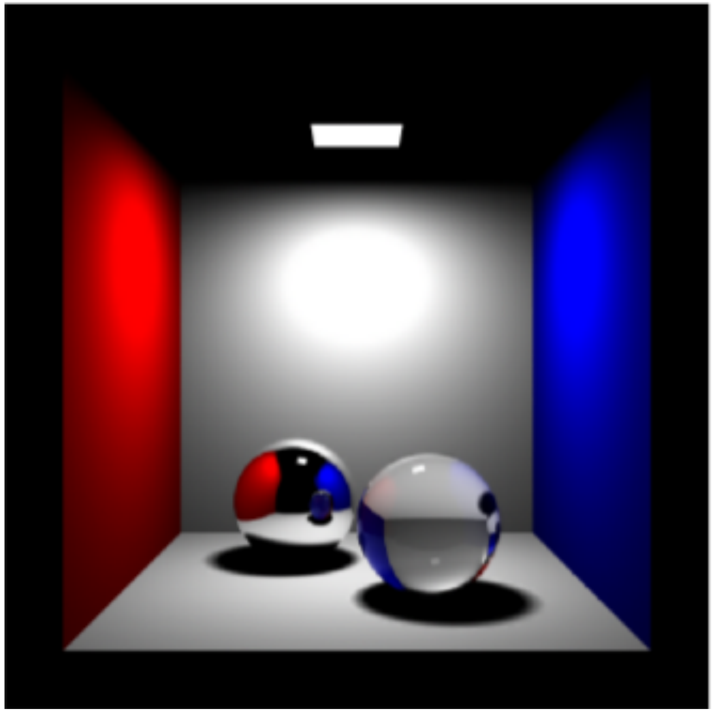
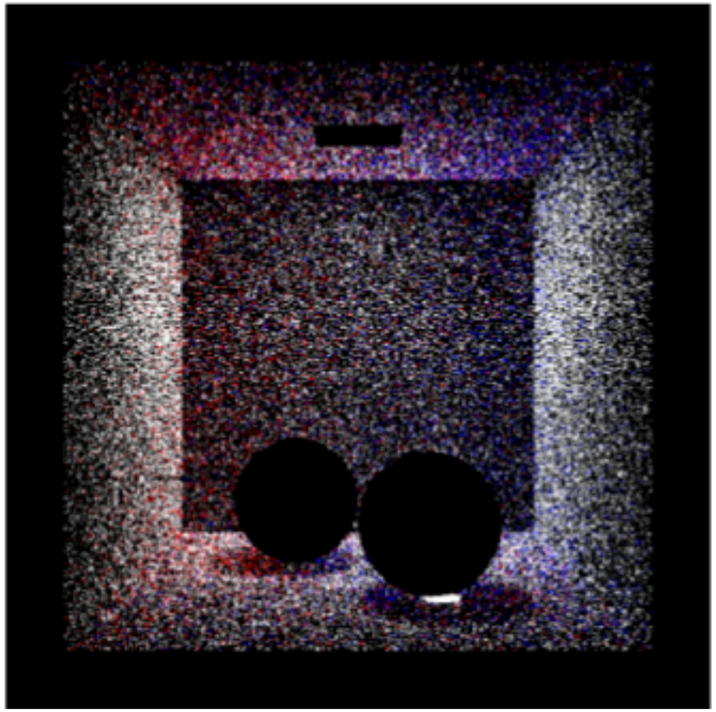
[CG Arena]

[deviantart]

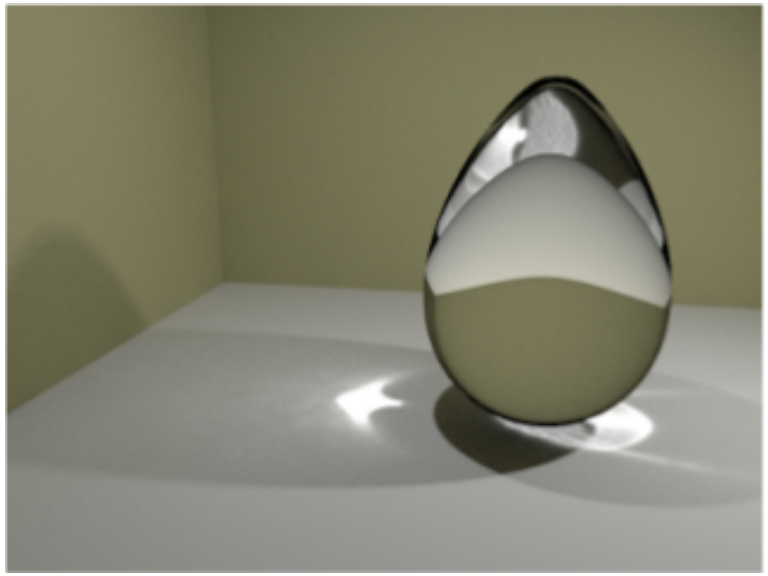
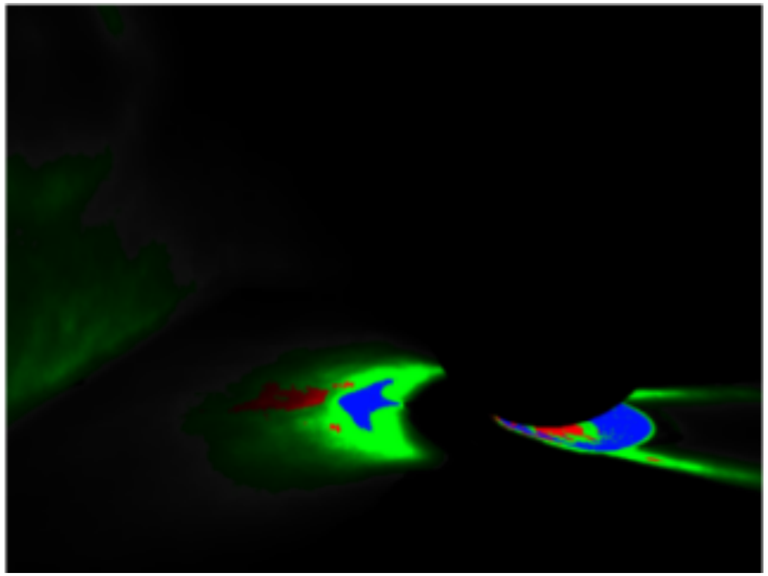
Utilisation d'une *photon map*



Caustics

carte de photons

Quantification des caustics

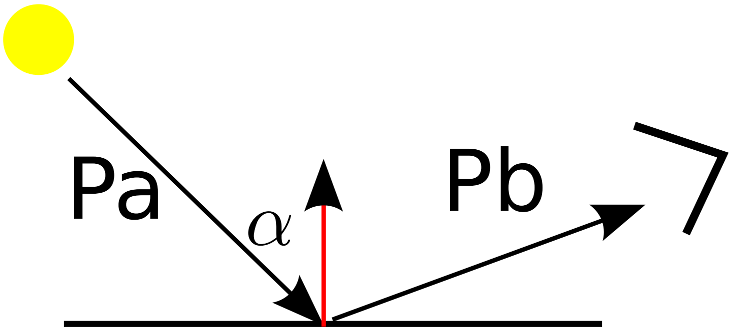
[Jensen, EGWR96]

Modèle physique

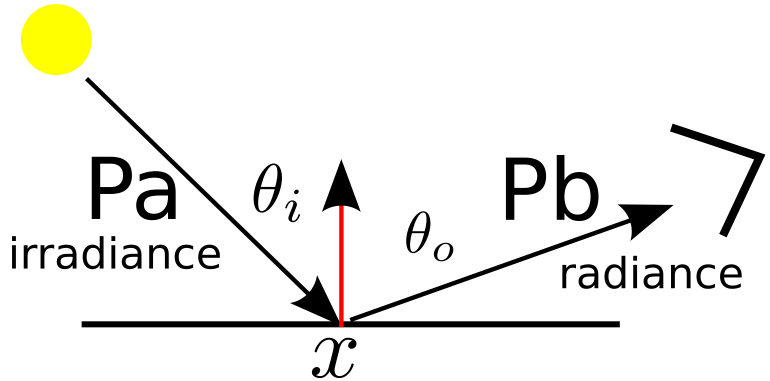
Physically based rendering

BRDF

Bidirectional
Reflectance
Distribution
Function



$$\cancel{P_b = \cos(\alpha)\, P_a}$$



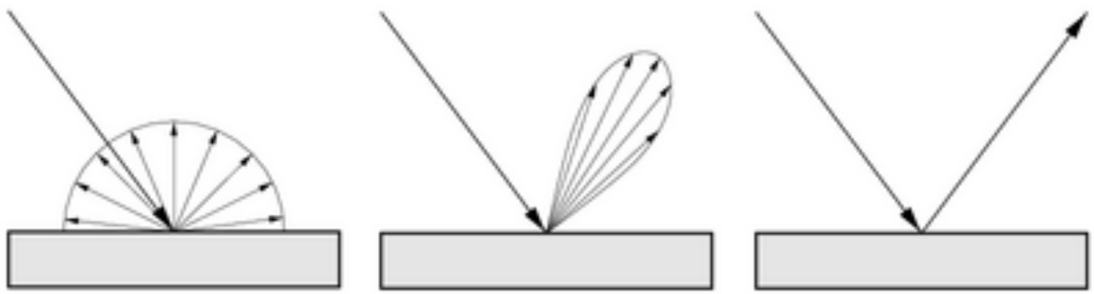
$$P_b = RBF(\theta_i, \theta_o, x)\cos(\theta_i)\, P_a$$

Si dépend de f => irridescence

[Wikipedia]

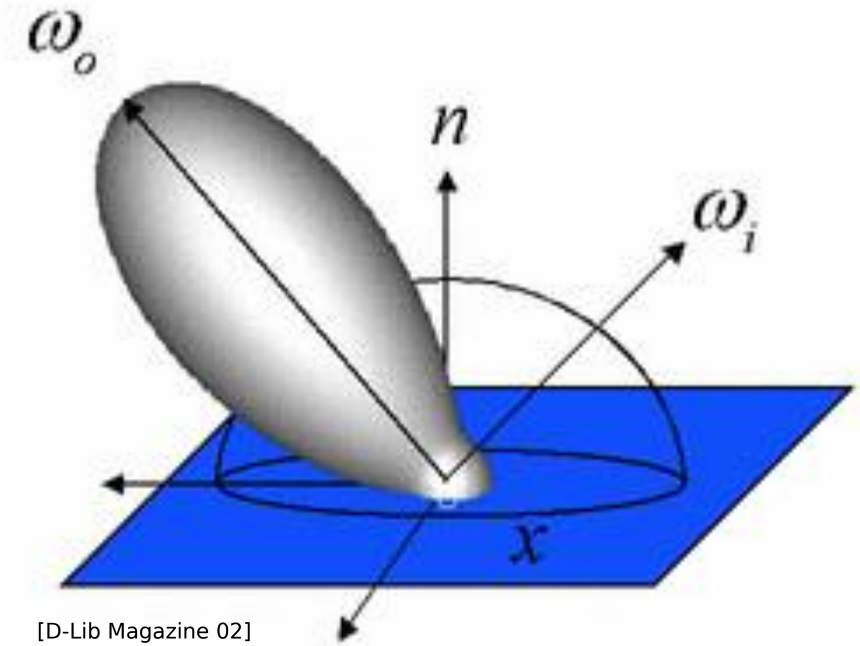
BRDF

Bidirectional **R**eflectance **D**istribution **F**unction



Diffus parfait

Speculaire parfais



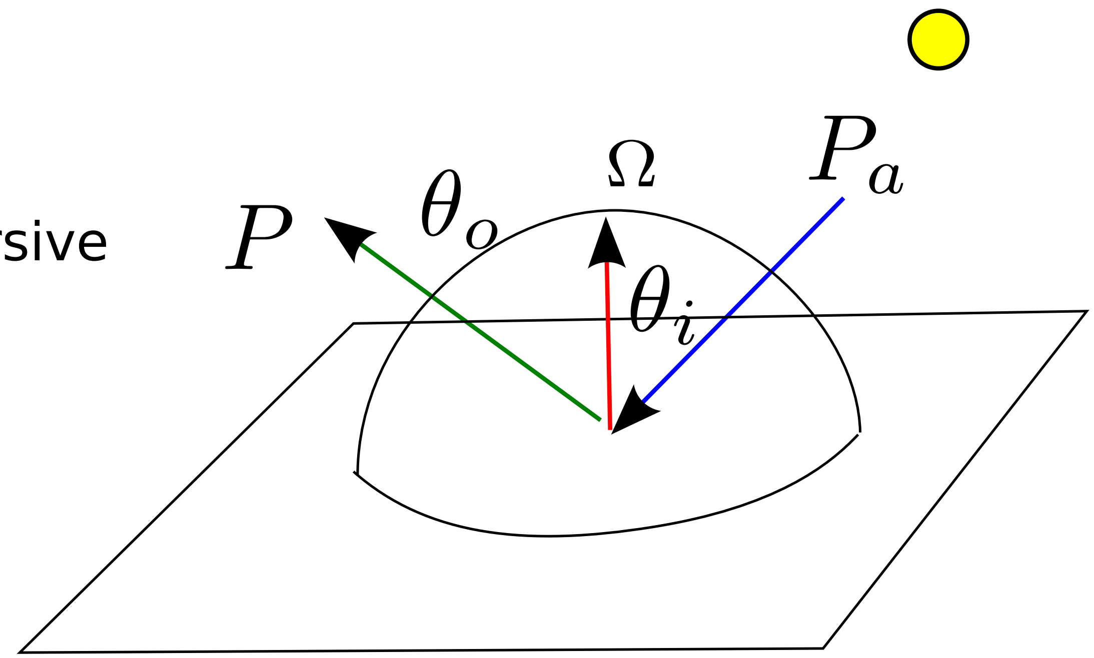
[D-Lib Magazine 02]

Equation de rendu

$$P(x,\theta_o) = P_e(x,\theta_o) + \int_{\theta_i \in \Omega} f(x,\theta_i,\theta_o)\, P_a(x,\theta_i) \cos(\theta_i)\, \mathrm{d}\theta_i$$

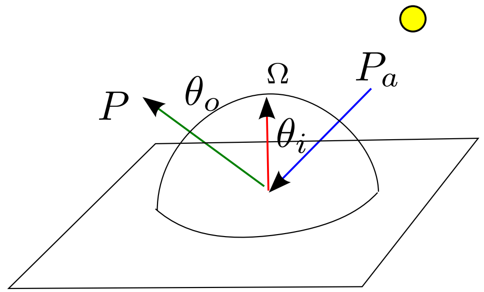
Problème: Pa dépend de P

Equation integrale infiniement recursive



Equation de rendu

$$P(x,\theta_o) = P_e(x,\theta_o) + \int_{\theta_i \in \Omega} f(x,\theta_i,\theta_o)\, P_a(x,\theta_i) \cos(\theta_i)\, \mathrm{d}\theta_i$$



2 Approches:

1- Discrétisation éléments finis

Radiosité

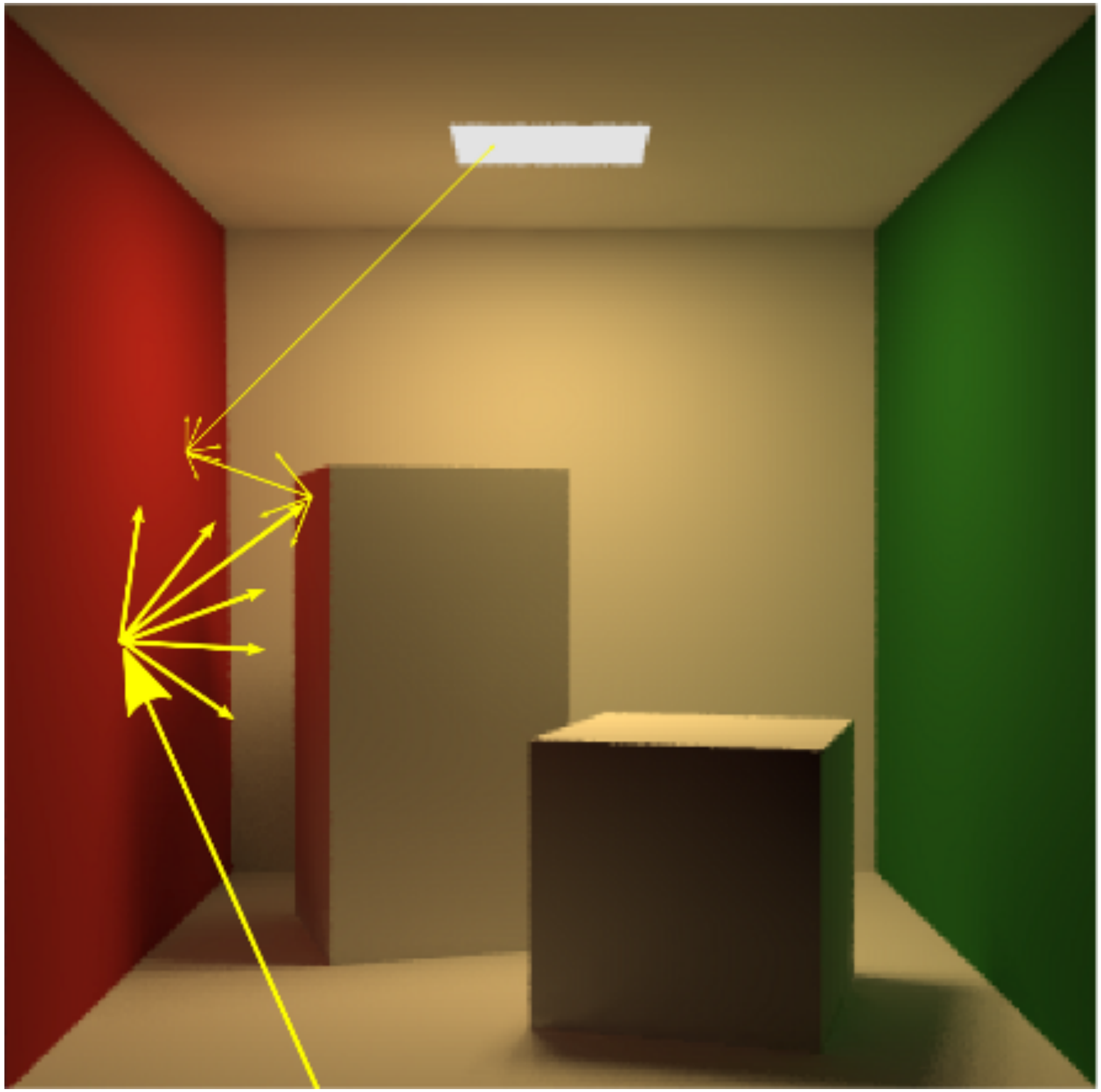
2- Monte-Carlo

Path Tracing

Metropolis Light Transport (MTL)

Path tracing

On échantillonne θ_i *au hasard*



Path tracing

Modélisation sources secondaire:

Illumination directe

Path tracing