En C++

OBJECT-ORIENTED RAY TRACING

Nathan CHAMPEIL & Laura MONTAGNIER

OBJECTIFS

OBJECTIF 1

Développer un environnement d'affichage

OBJECTIF 2

Utiliser une programmation objet

OBJECTIF 3

Ajouter des options : couleurs, matériaux, etc...

SOMMAIRE

TRACEUR BASIQUE

Classes choisies, lumière, calcul des intersections...

AMÉLIORATION

Diffusion de la lumière, couleur, matériau...

ALLER PLUS LOIN...

Optimisation & nos choix personnels d'approfondissement

1) TRACEUR BASIQUE

LES VOLUMES

Dans volumes.hpp

- Le point
- La sphère
- Le plan

La sphère

3 coordonnées du centre
= un vecteur
Un rayon
= un *double*

Le point

3 coordonnées
=un vecteur

Le plan

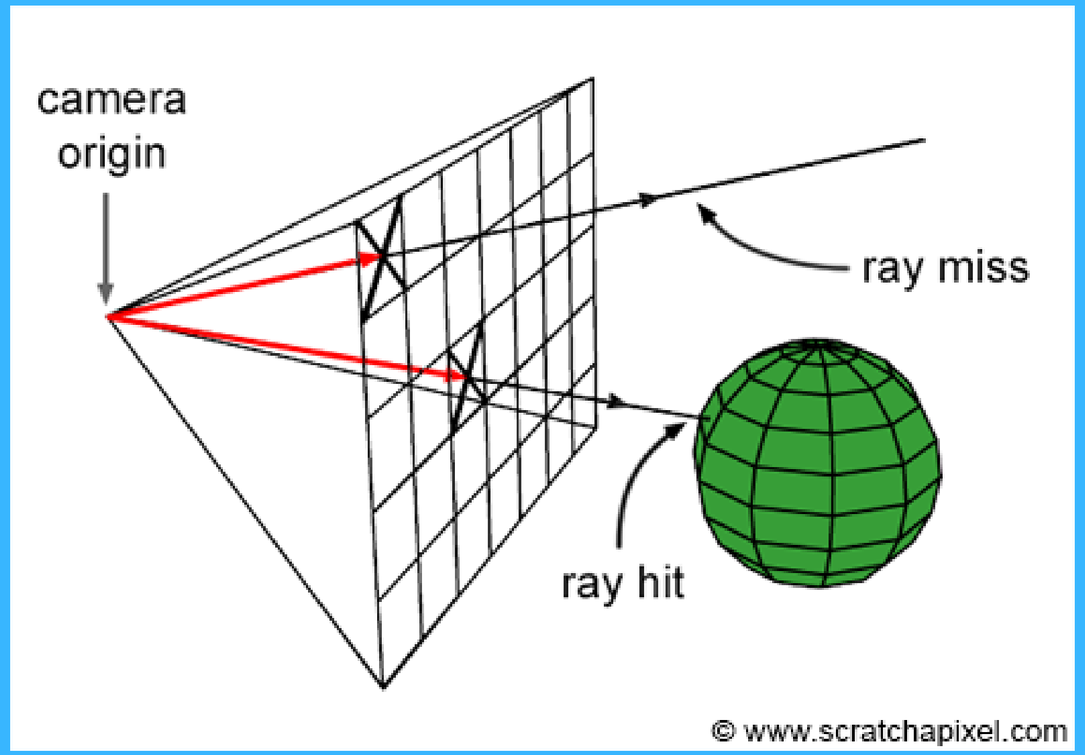
3 coordonnées d'un
point = un vecteur
Un vecteur normal
= un vecteur

1) TRACEUR BASIQUE

INTERSECTIONS

Dans volumes.cpp

- La sphère
- Le plan



... toujours avec un rayon !

1) TRACEUR BASIQUE

RAYON-SPHÈRE

- On écrit le rayon sous la forme O + tD, avec O son origine qui sert aussi d'origine au repère, D sa direction, et l'instant t.
- La sphère de centre (x_C, y_C, z_C) et de rayon R a pour équation :

$$(x - x_C)^2 + (y - y_C)^2 + (z - z_C)^2 = R^2$$

- On cherche l'intersection entre le cercle et le rayon qui se fait à l'instant t. Pour cela, on veut satisfaire l'équation :

$$||O + tD - C||^2 = R^2$$

- On résoud :

$$(O + tD - C).(O + tD - C) = R^2$$

$$t^2 D.D + 2tD.(O - C) + O.O + C.C - 2OC = R^2$$

- On note :

$$a * t^2 + b * t + c = 0$$

avec

$$a = D^2, b = 2(O - C)D, c = ||O - C||^2 - R^2$$

- Les solutions dépendent donc du signe du discriminant : $\delta = b^2 - 4ac$. Si $\delta < 0$, il n'y a pas de solution donc pas d'intersection entre le rayon est la sphère. Si $\delta = 0$, il y a précisément une seule intersection en $t_1 = \frac{-b}{2a}$. Si $\delta > 0$, il y a deux intersections et il suffit de trouver laquelle est la plus proche de l'origine.

1) TRACEUR BASIQUE

RAYON-PLAN

Intersection rayon-plan :

$O, v, p, n \in R^3$

Rayon : $O + tv$ paramétré par $t \in R_+$

Équation du plan de normale n passant par p :

$x | (x - p).n = 0$

Résolution : $0 = (O + tv - p).n = tv.n + (O - p).n$

Donc $t = \frac{(p-O).n}{v.n}$

1) TRACEUR BASIQUE

QUID DE LA LUMIÈRE ?

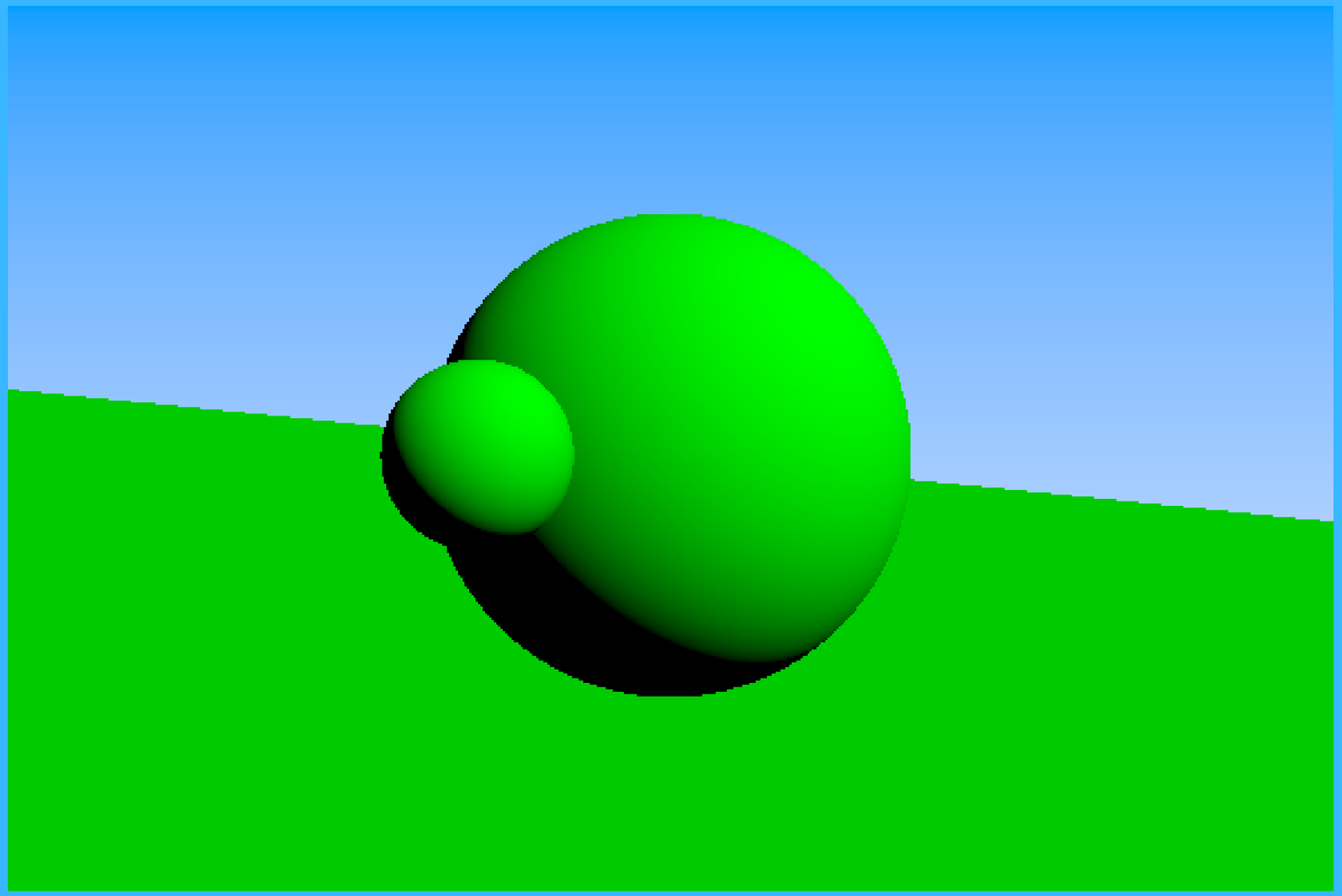
SOURCE
À
L'INFINI

UN RAYON
= une classe

Intersections
colorées

1) TRACEUR BASIQUE

NOTRE PREMIÈRE IMAGE



2) AMÉLIORATION

REBONDS

Tout dans rebond.cpp

Combien de types de rebonds ?

-Diffusion

-Réflexion

-Réfraction

2) AMÉLIORATION

MATÉRIAUX

Inclus dans volumes.cpp

6 champs

Tous entre 0 et 1 :

-RGB (3 premiers champs)

-Probabilité de réflexion

-Probabilité de réfraction

-Lumineux (booléen)

2) AMÉLIORATION

PRINCIPE

graphics.cpp

Utilise SDL

Comment fonctionne camera.cpp ?

Définir les matériaux

Définir les objets

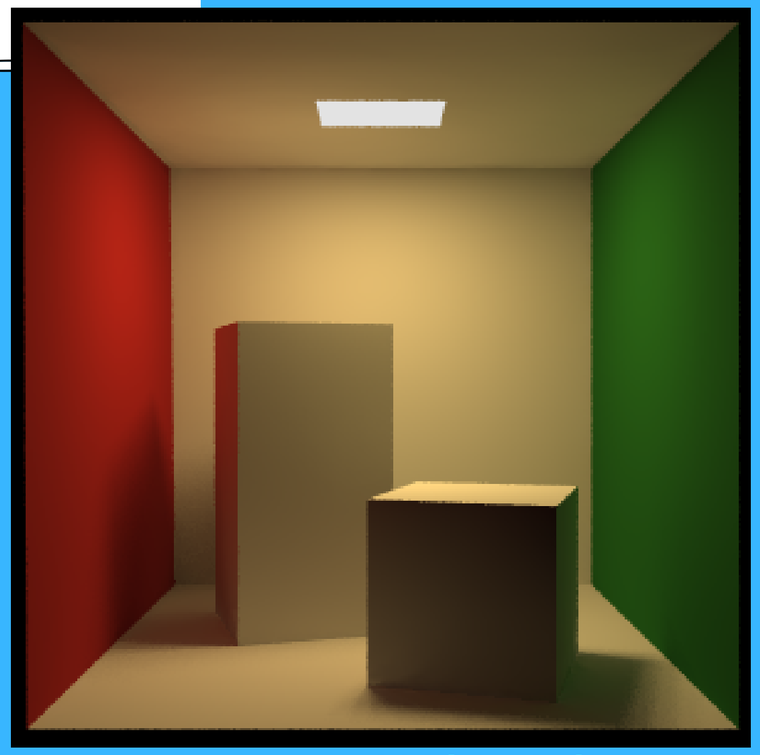
Définir leur union

Envoyer les rayons

Coloriser les pixels

2) AMÉLIORATION

Objectif :



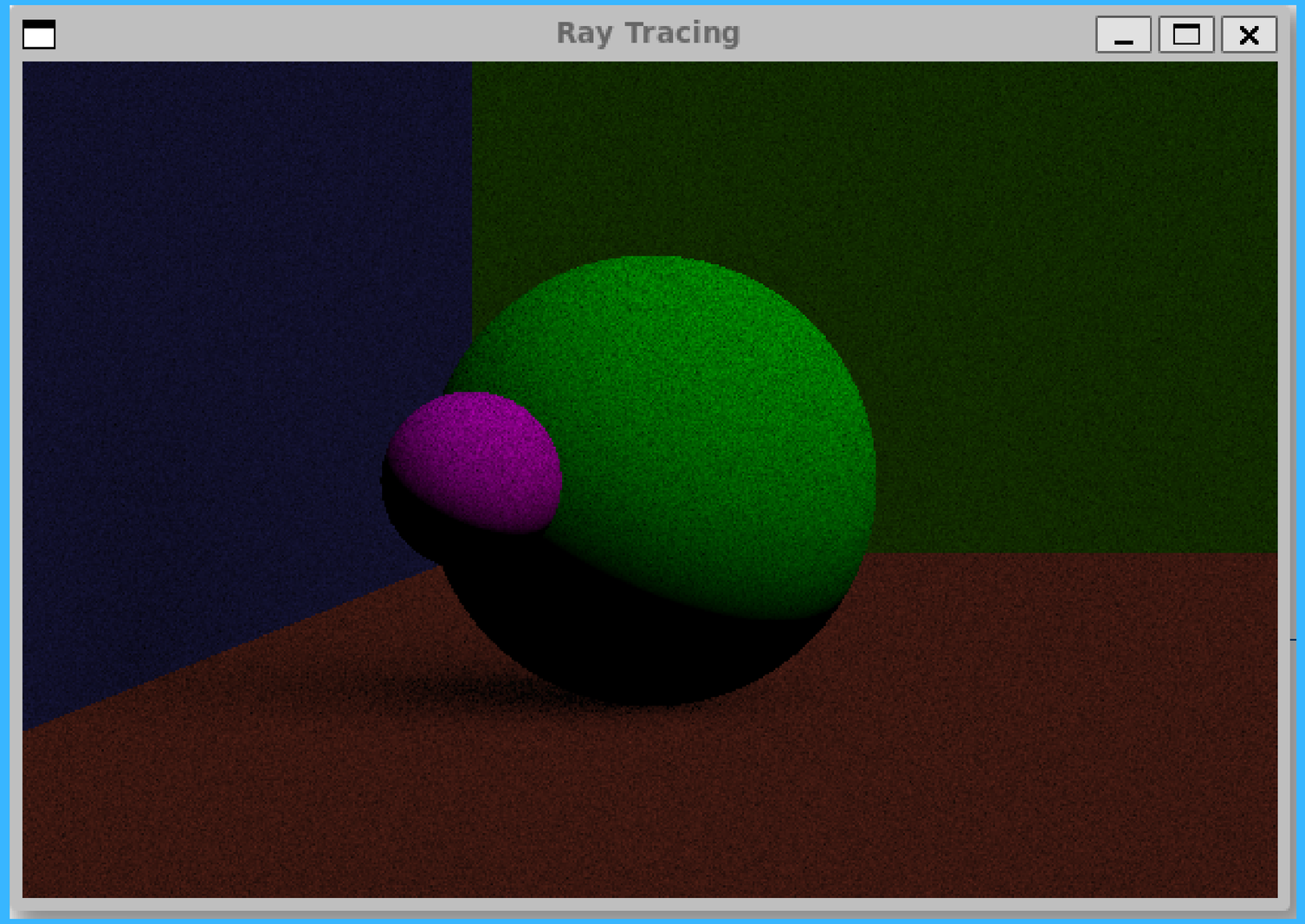
Wikipédia.com

LA BOITE DE CORNELL

Concept d'imagerie de synthèse informatique

2) AMÉLIORATION

Problème :
la lumière

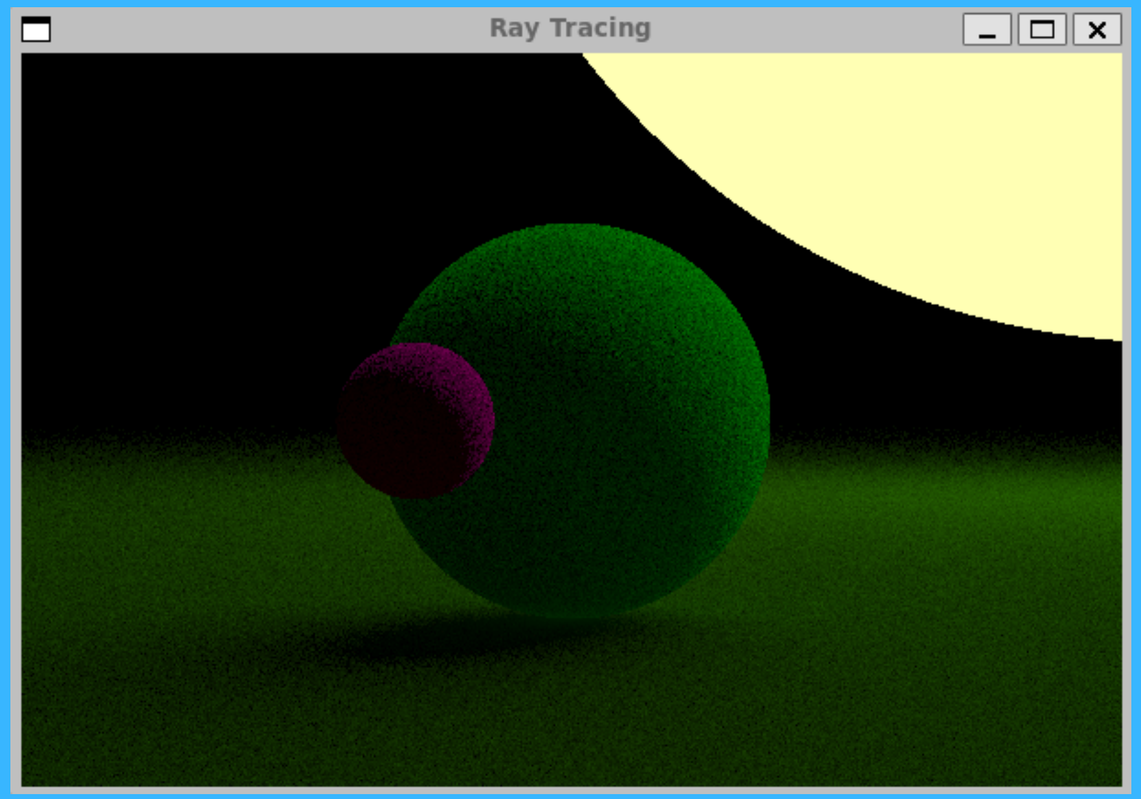


LA BOITE DE CORNELL

SOLUTION : OBJETS LUMINEUX...

3) ALLER PLUS LOIN

OBJETS LUMINEUX



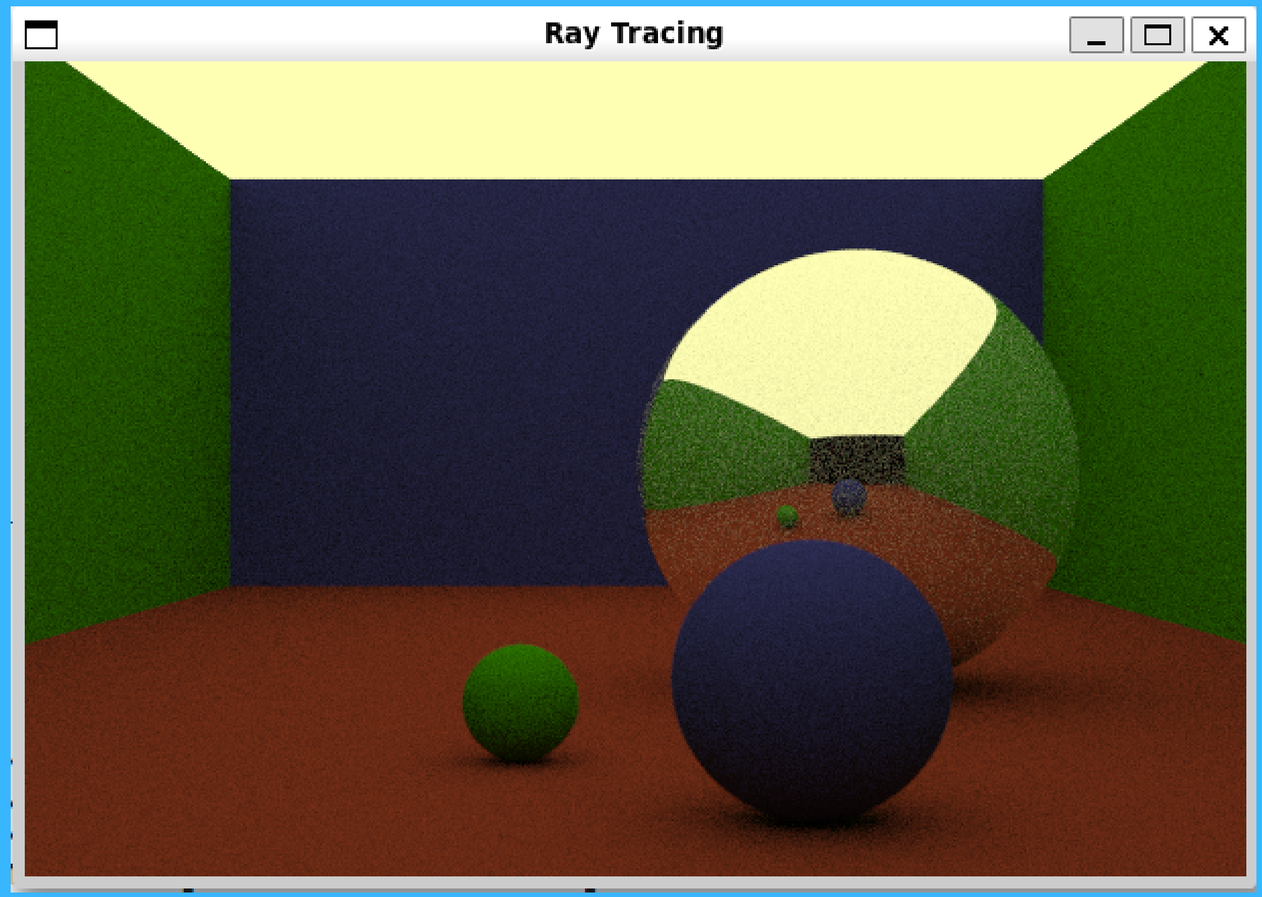
FONCTIONNEMENT

Un rayon qui “tape” dans un objet lumineux s’arrête.

3) ALLER PLUS LOIN

PLAFOND LUMINEUX

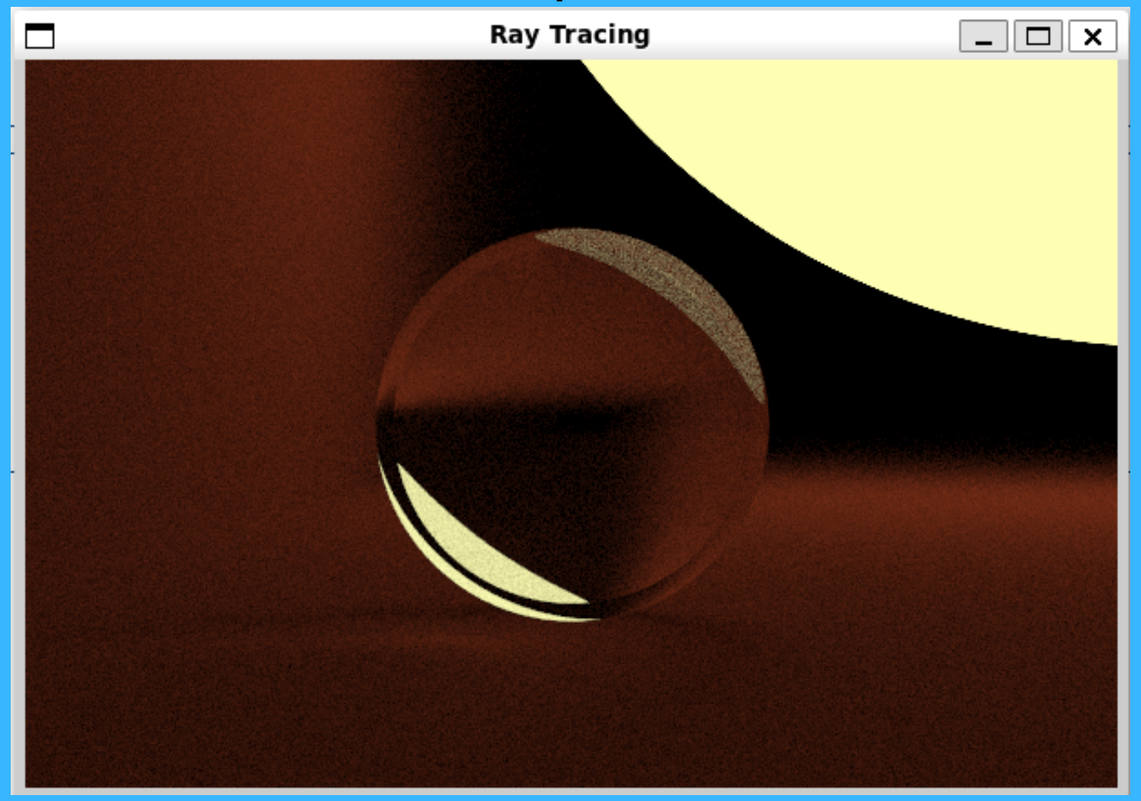
La grosse sphère est un miroir !



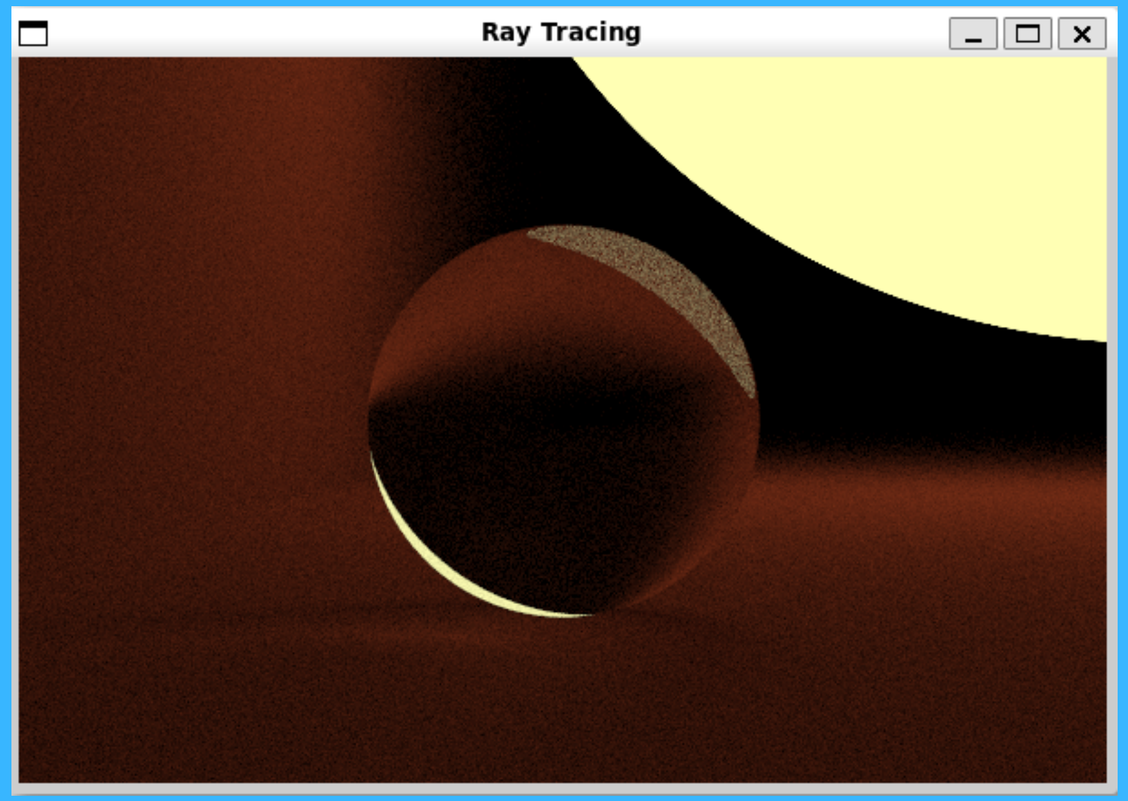
3) ALLER PLUS LOIN

LA TRANSPARENCE

Wikipédia

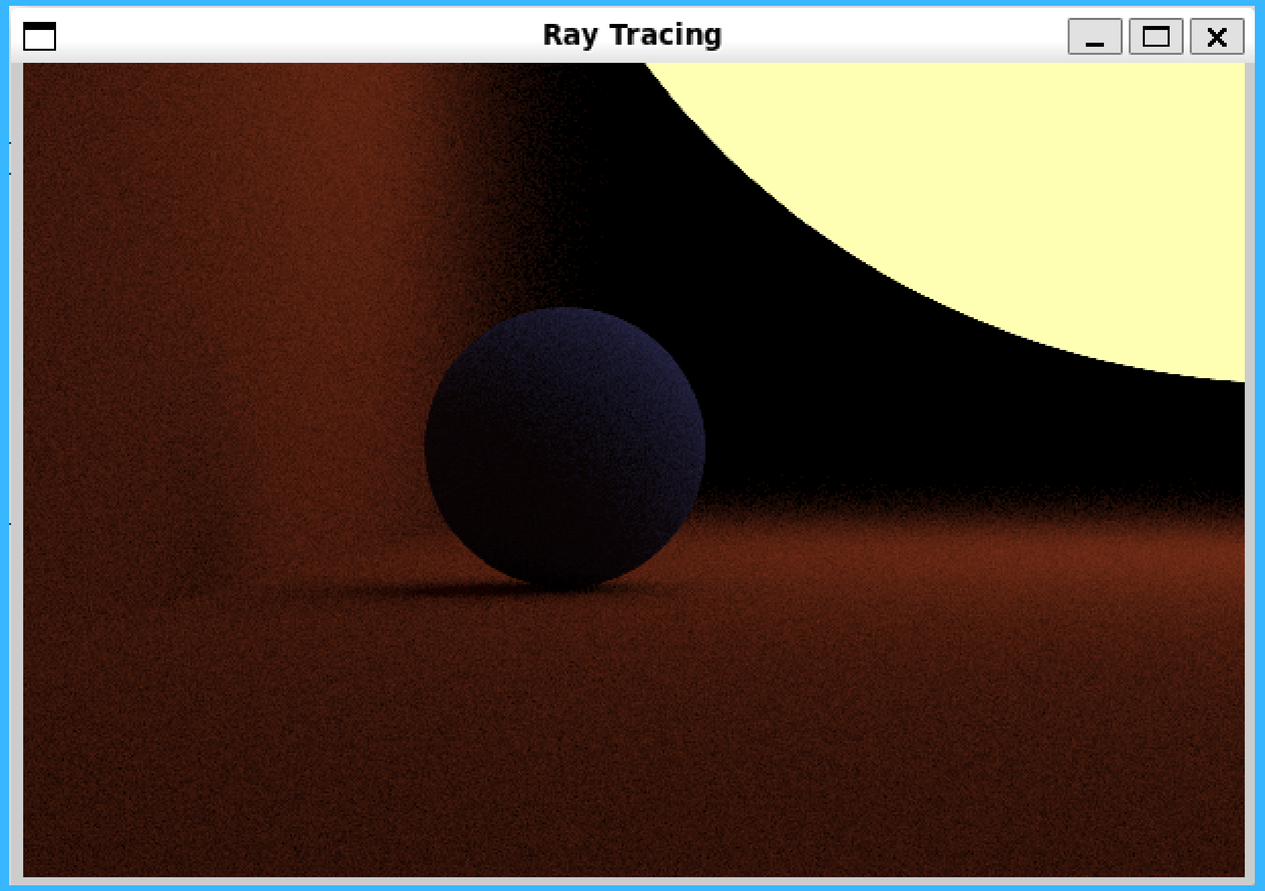


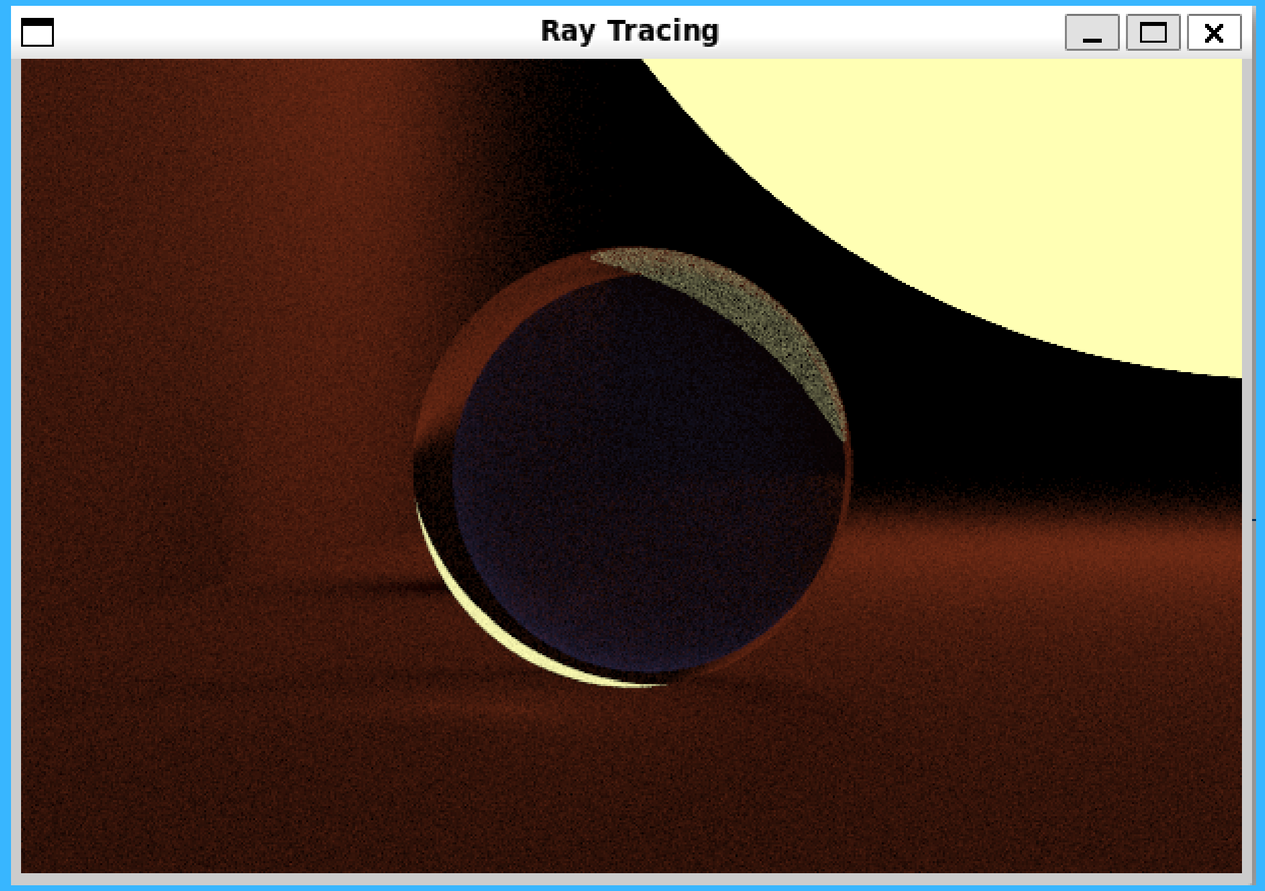
Nous



3) ALLER PLUS LOIN

VERRE



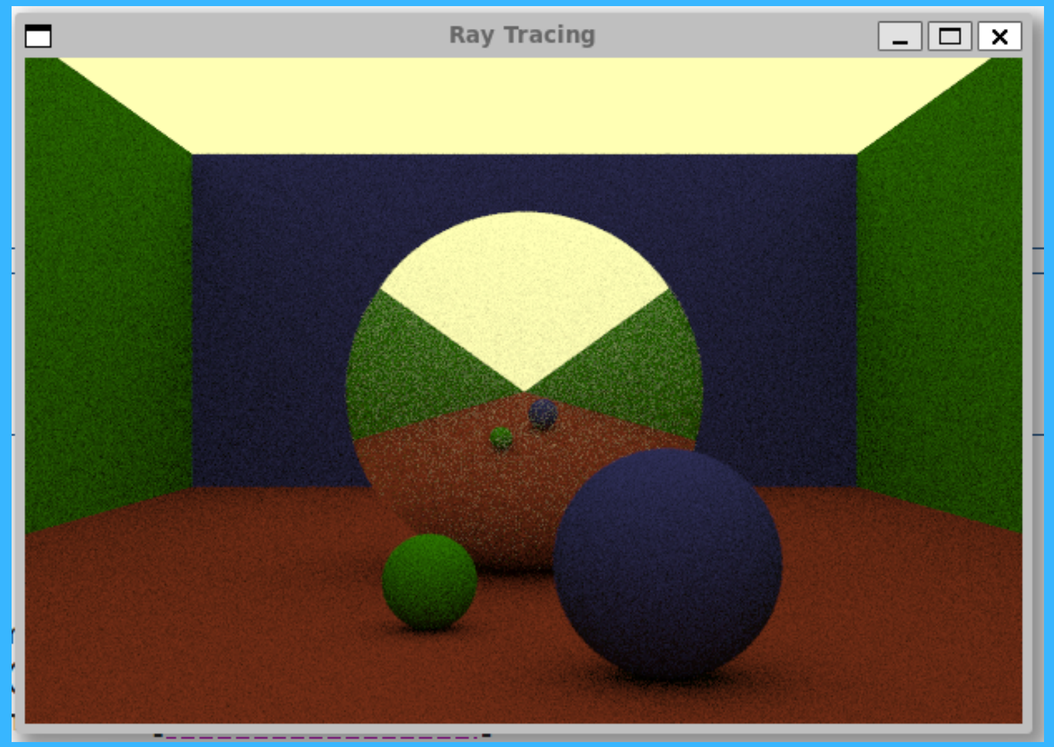


On a même l'effet loupe !

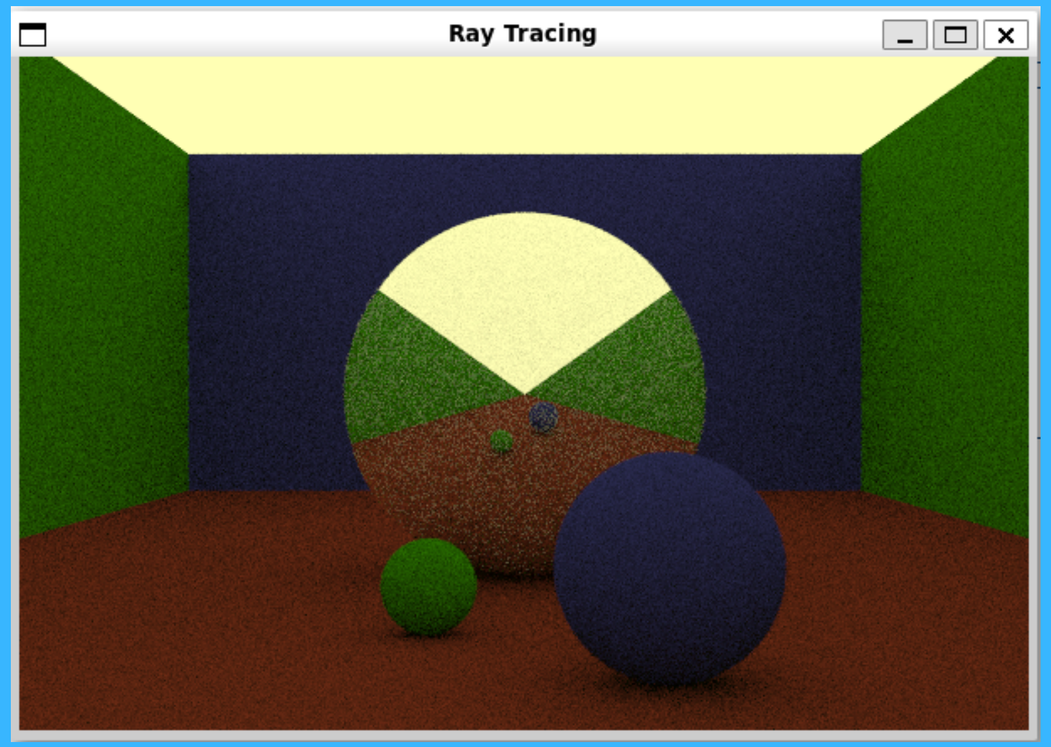
3) ALLER PLUS LOIN

QUESTIONNER NOS CHOIX

En utilisant une demi-sphère aléatoire pour la diffusion, un cosinus et la méthode du rejet



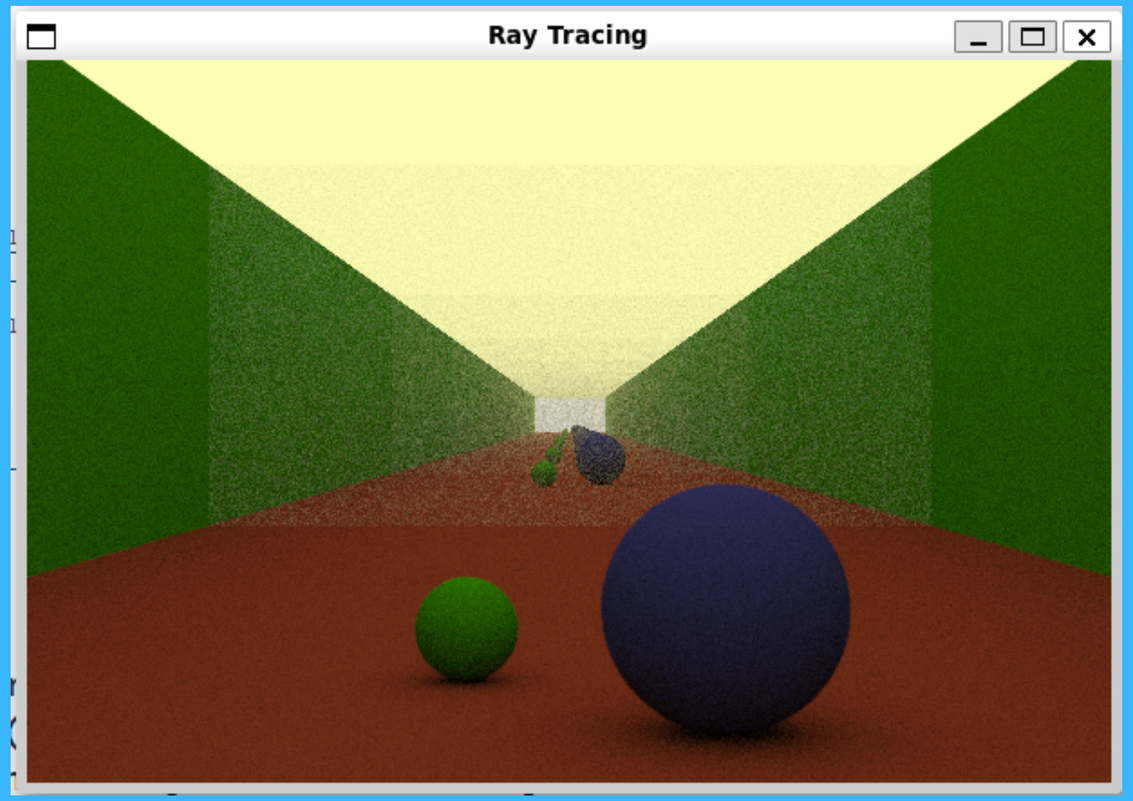
En utilisant une sphère tangente pour la diffusion



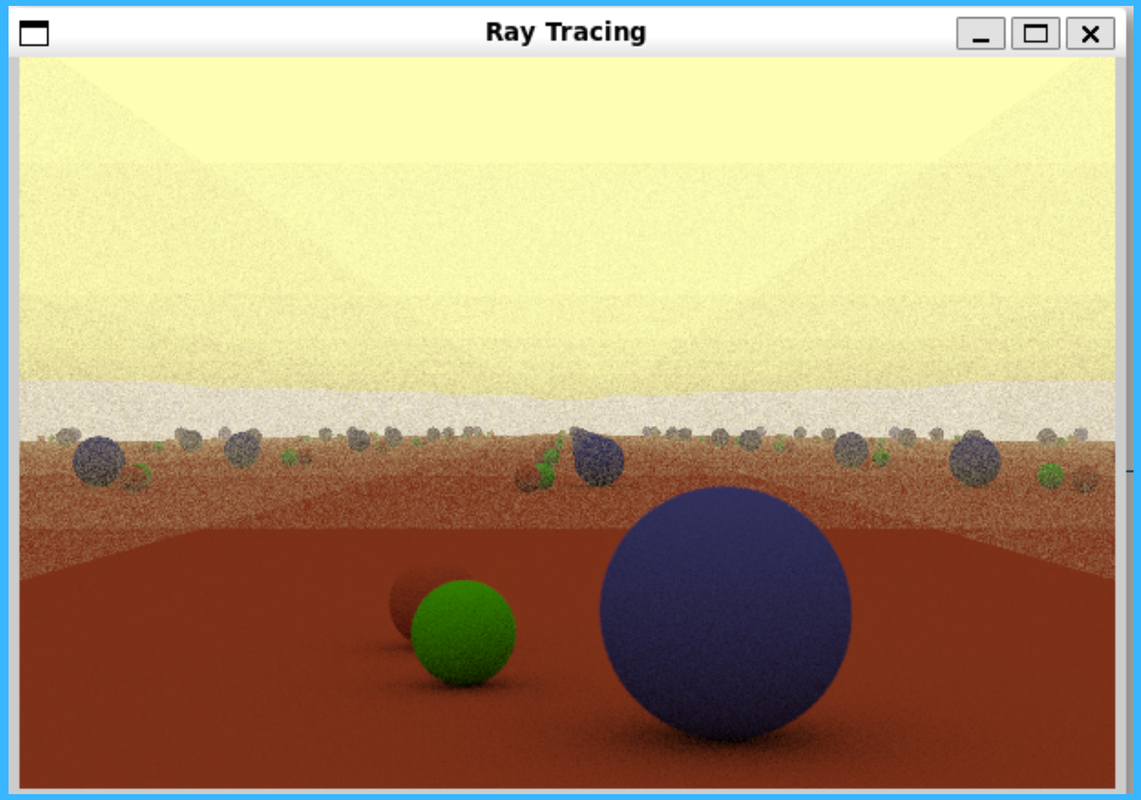
3) ALLER PLUS LOIN

INTÉRÊT DE L'IMAGERIE

Deux miroirs



4 miroirs



3) ALLER PLUS LOIN

PARALLÉLISME

...pour aller plus vite

MÉTHODE

Utilisation de OpenMP pour répartir
les lignes de pixels entre les processeurs.
(Cela se fait automatiquement.)

SPEED-UP

Mesure : de 11 secondes à 5 secondes
Accélération de 2,2 !

CONCLUSION

NOS DIFFICULTÉS

Sur quoi a-t-on eu du mal ?

A RETENIR

Qu'est-ce que nous avons appris ?

3) ANNEXES

ARBORESCENCE DE FICHIERS

camera.hpp

Lance les rayons

main.hpp

Appelle les fonctions

defs.hpp

Sert de hub pour toutes les inclusions

volumes.cpp

Définit les classes, les calculs de vecteurs, et les intersections

rebond.hpp

Rebond du rayon de lumière en diffusion de Lambert

init.hpp

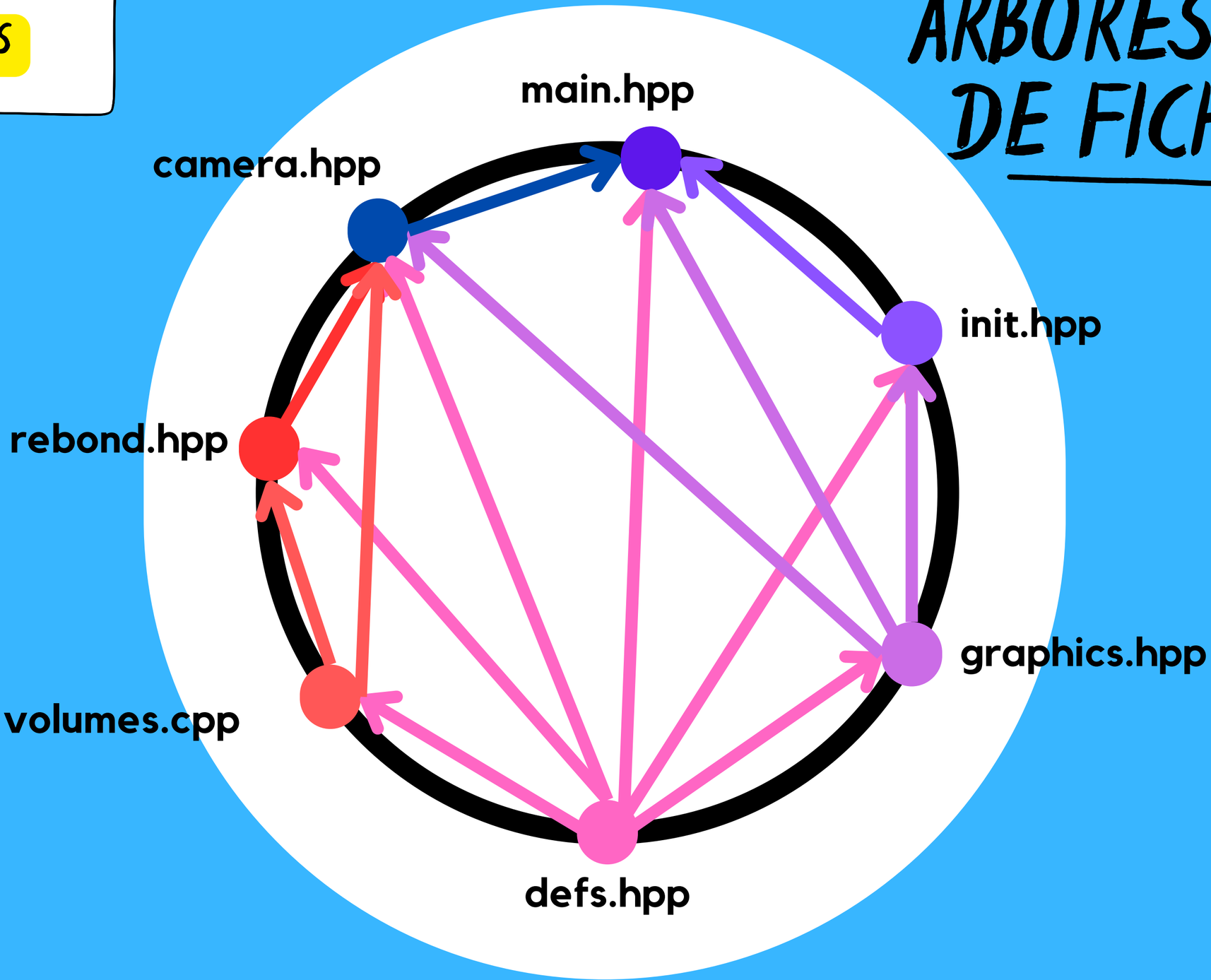
Initialise tout ce qui concerne SDL et ferme SDL

graphics.hpp

Utilise SDL pour gérer la fenêtre

3) ANNEXES

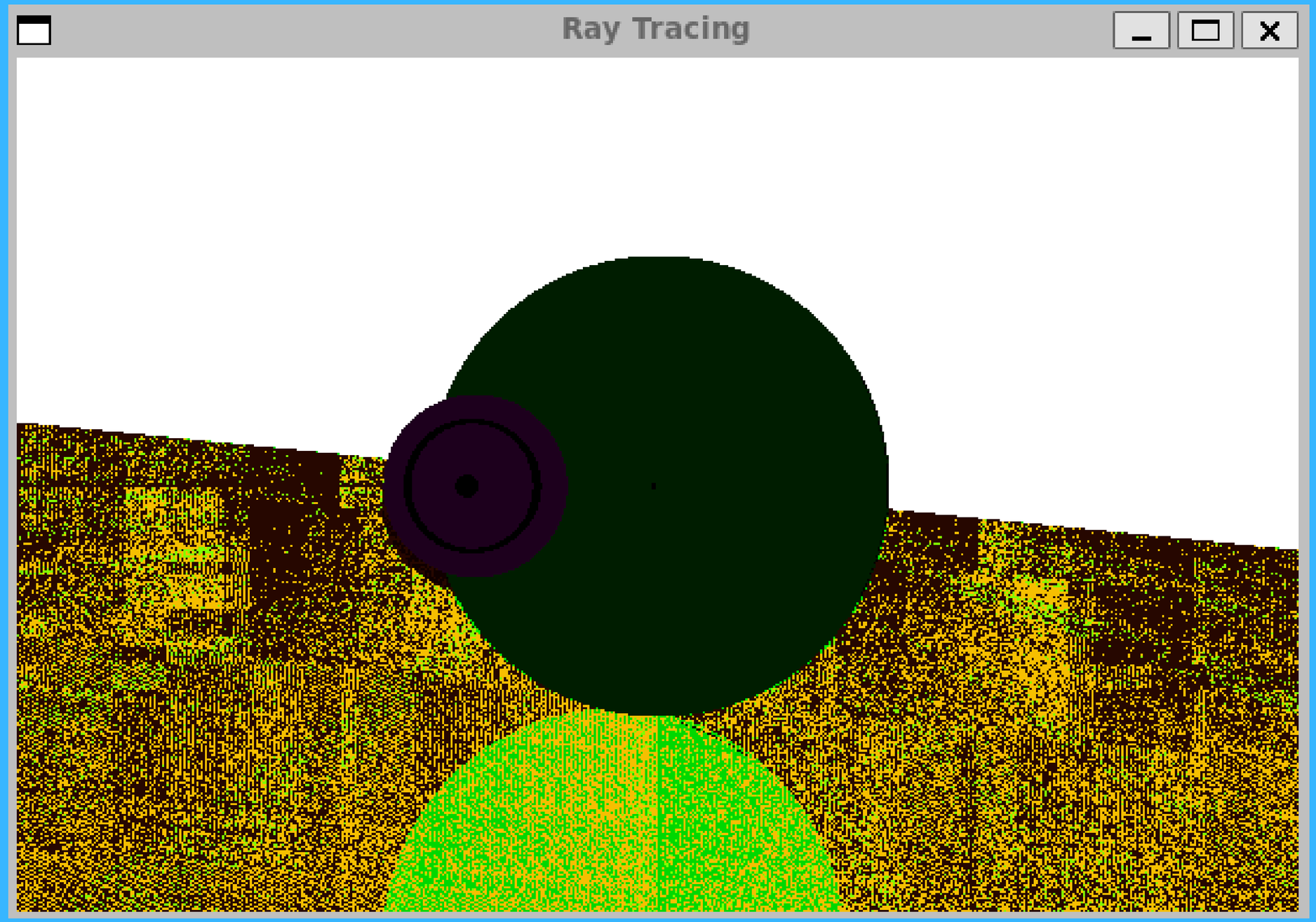
ARBORESCENCE DE FICHIERS



3) ANNEXES

Problème 1 :

Lorsque les RGB pouvaient dépasser 255



3) ANNEXES

Problème 2 :

Lorsque les rayons se trouvaient bloqués du mauvais côté de la paroie