

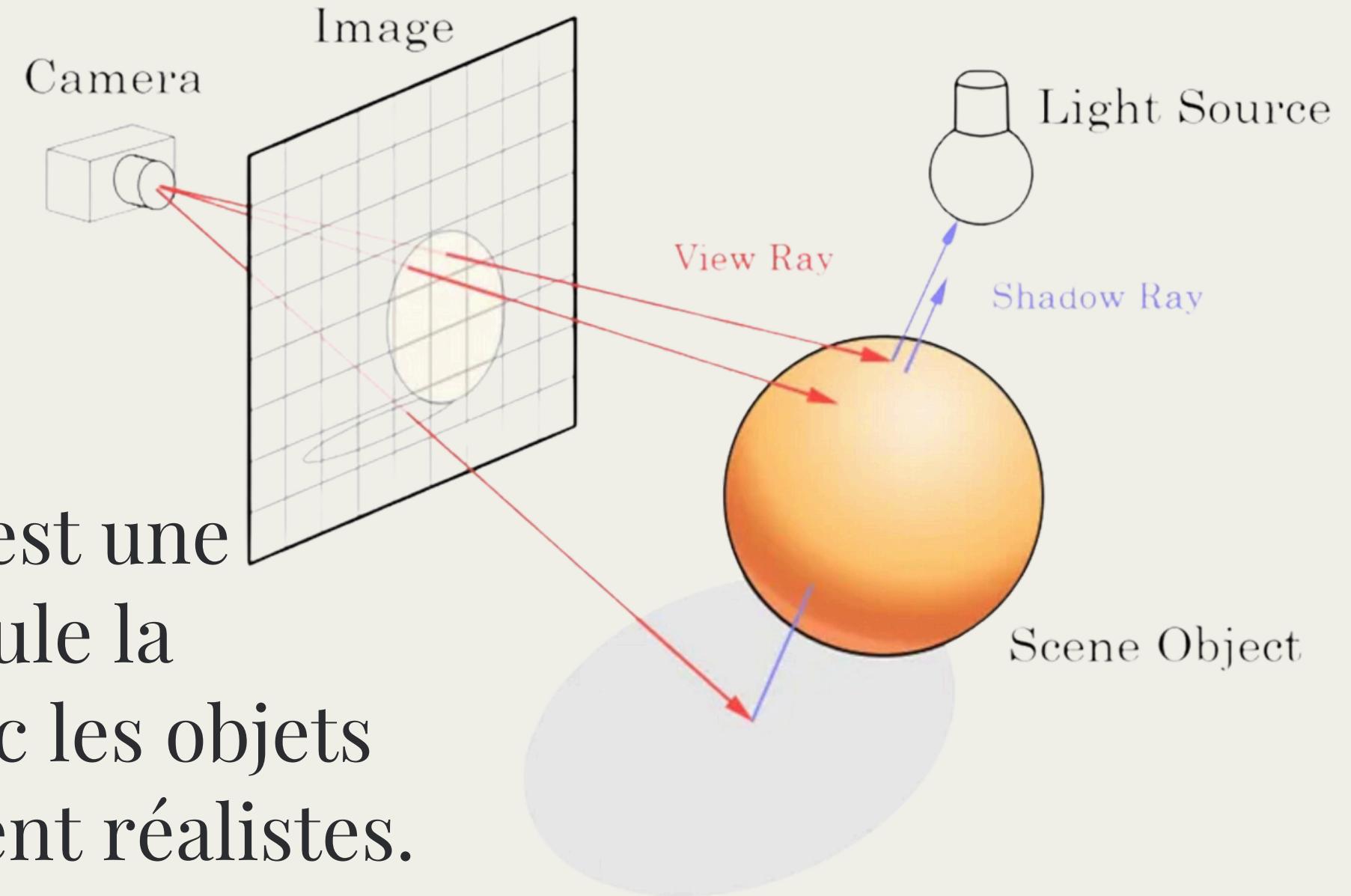
# Lancer de Rayons

---

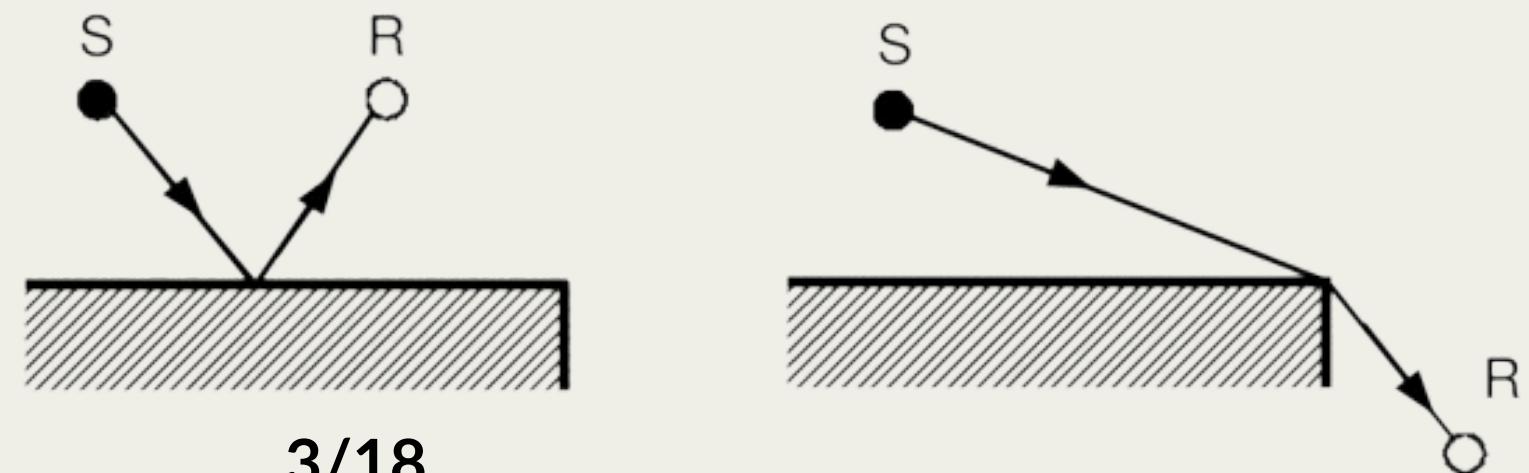
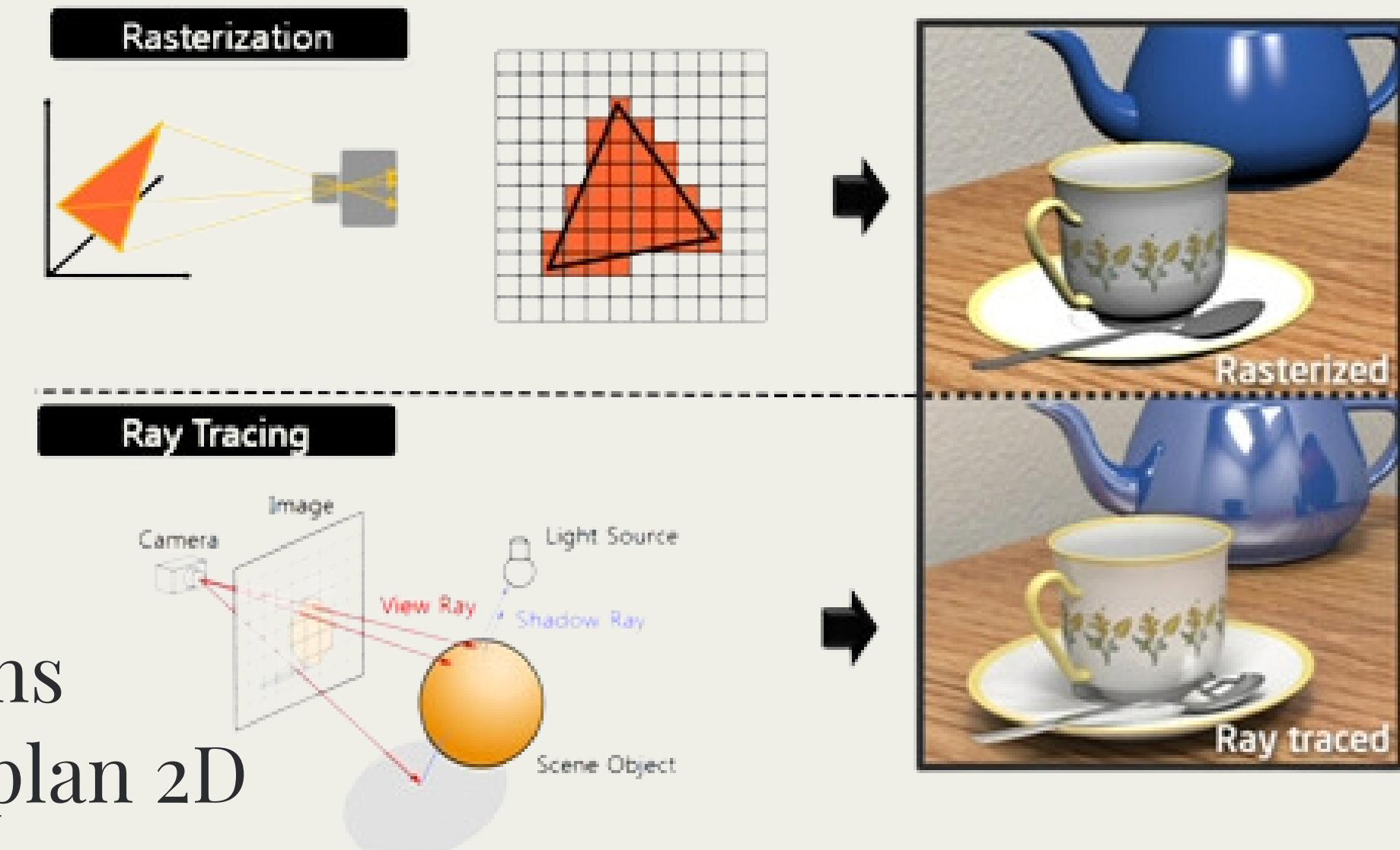
ORIENTÉ OBJET

MZIOU MOHAMED  
BEN AISSA TAKOUA  
28/01/2025

Le Ray Tracing, ou lancer de rayons, est une méthode de rendu graphique qui simule la manière dont la lumière interagit avec les objets pour produire des images extrêmement réalistes. Cette technique repose sur des calculs mathématiques précis pour tracer le chemin des rayons lumineux, depuis leur source jusqu'à leur interaction avec des objets dans une scène virtuelle.

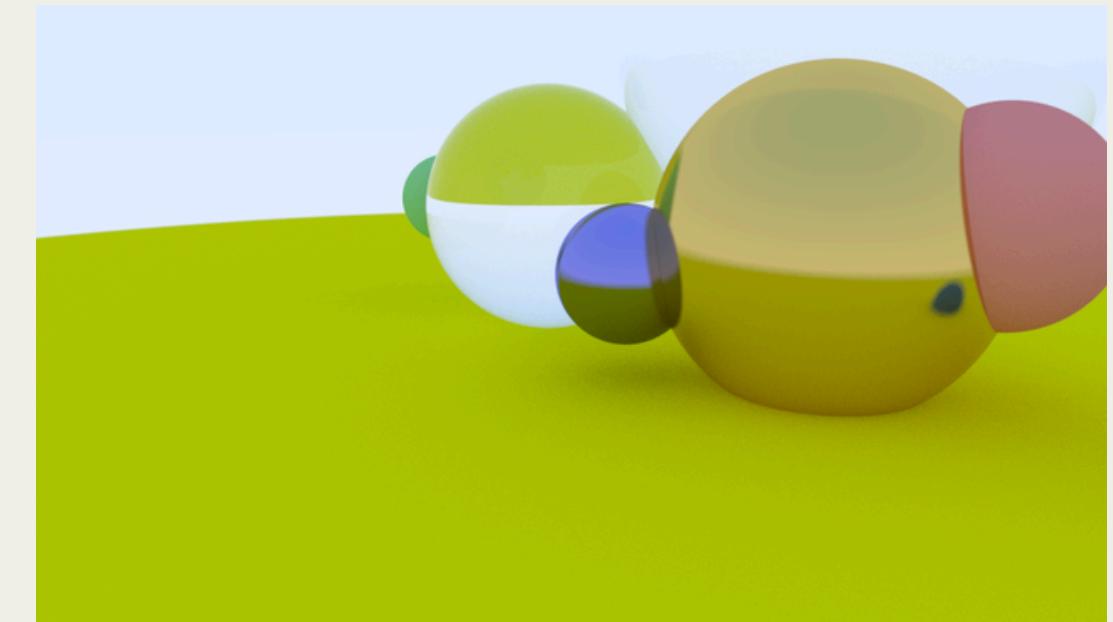
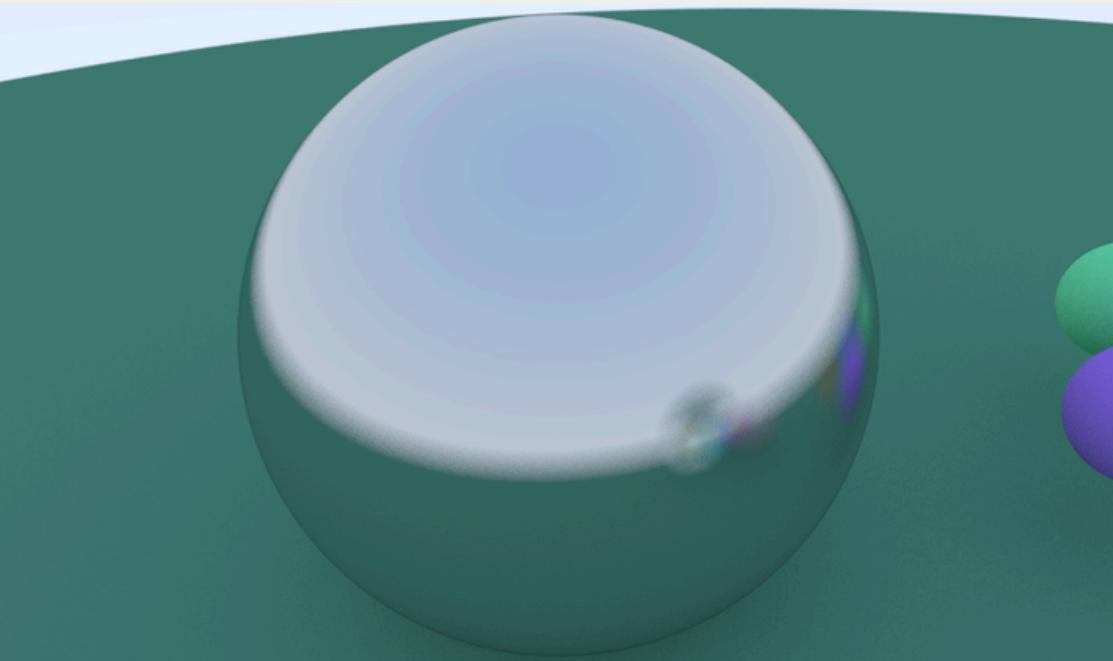
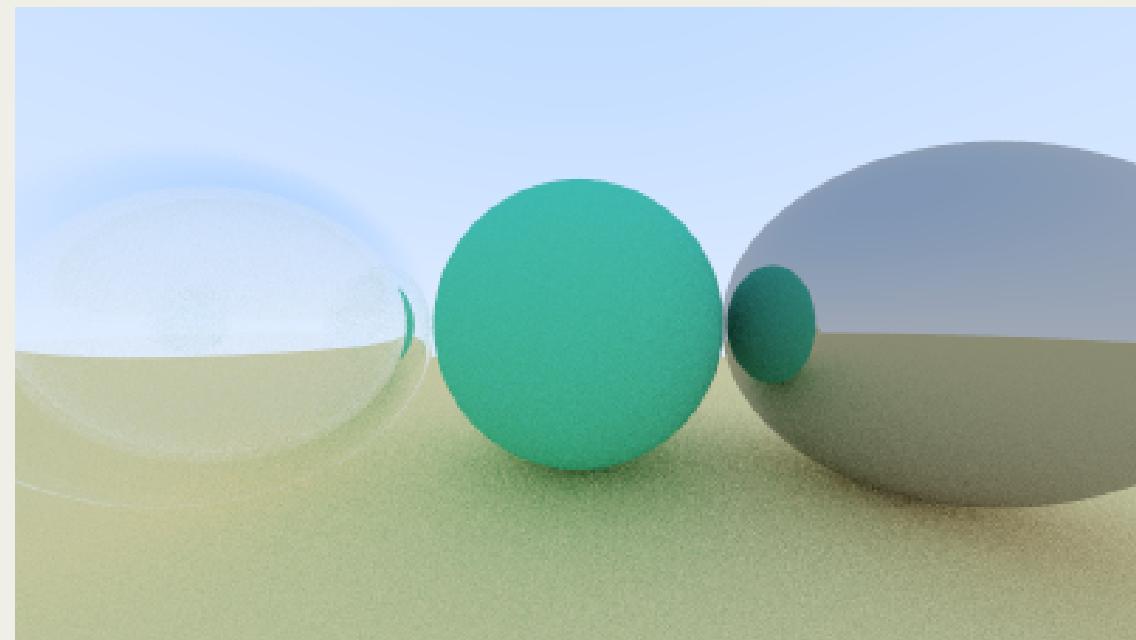
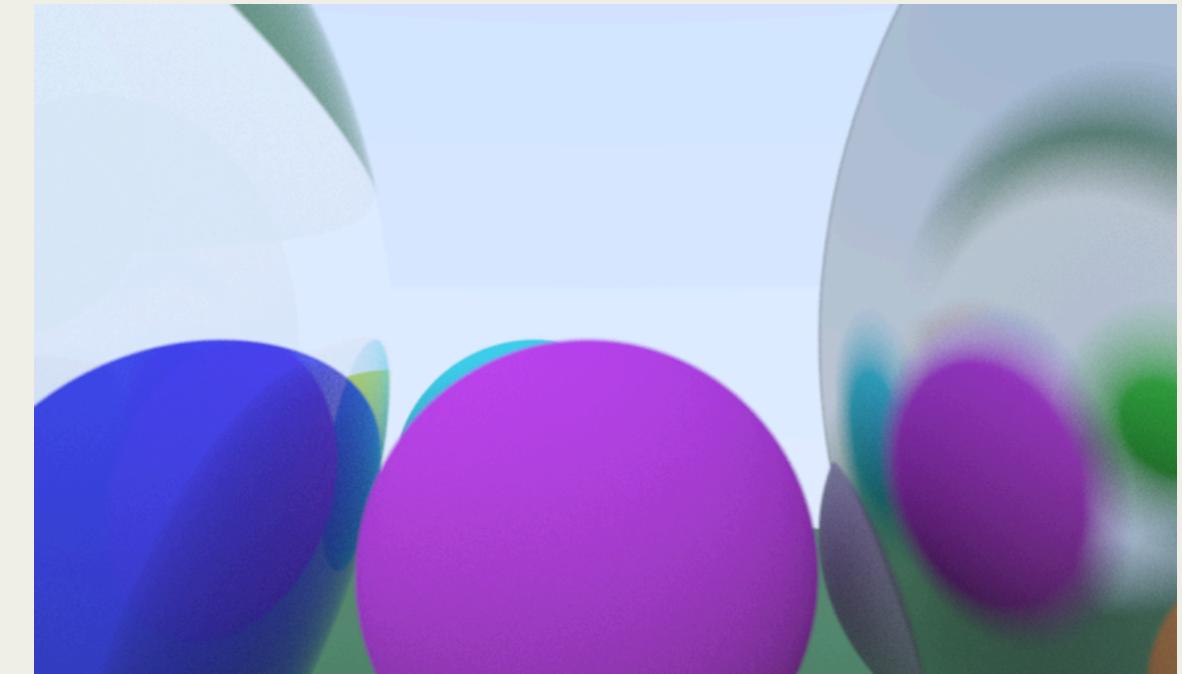
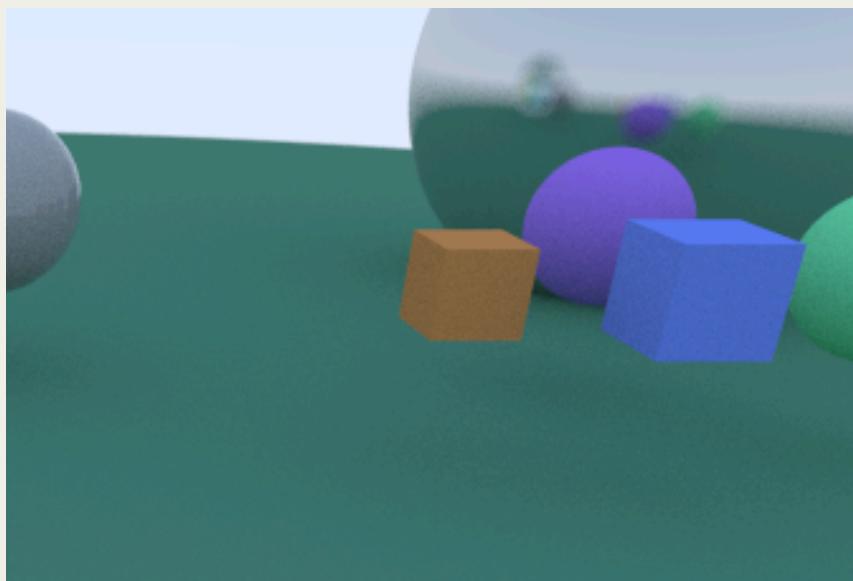
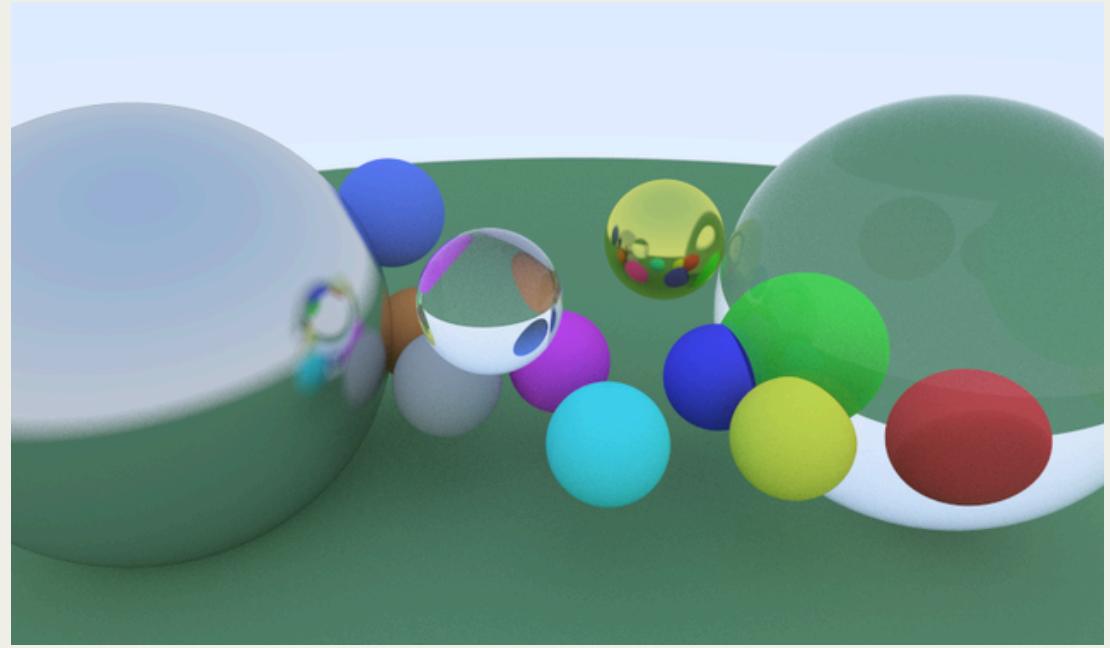


Contrairement à la méthode classique (**rasterization**), qui simplifie les interactions lumineuses en projetant les objets sur un plan 2D à l'aide de triangles, le Ray Tracing offre une approche plus réaliste en simulant les réflexions, réfractions et diffusion avec une précision physique remarquable.



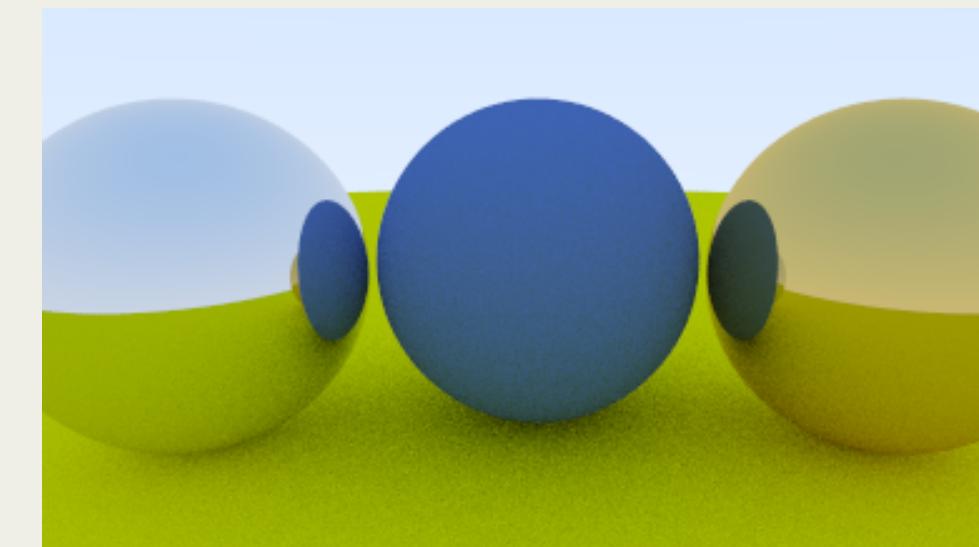
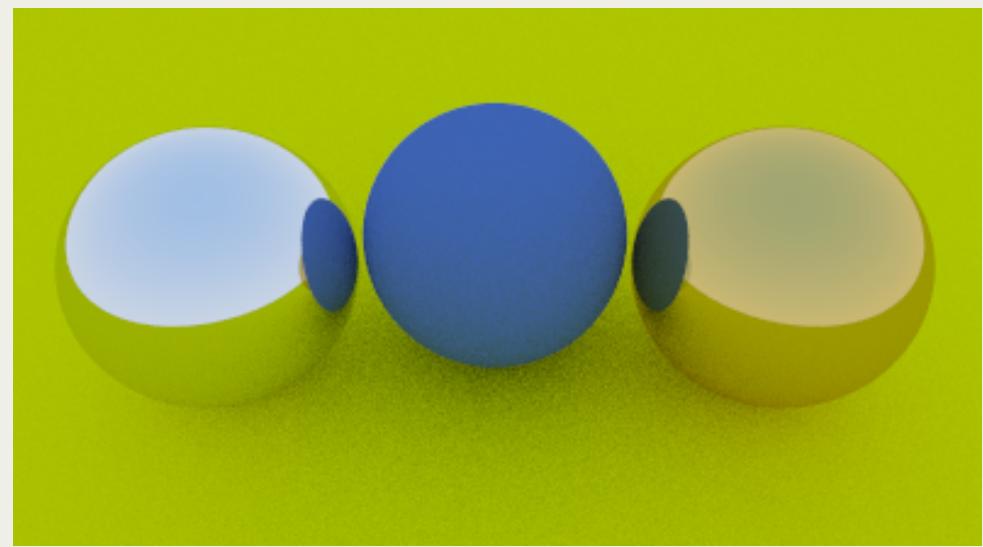
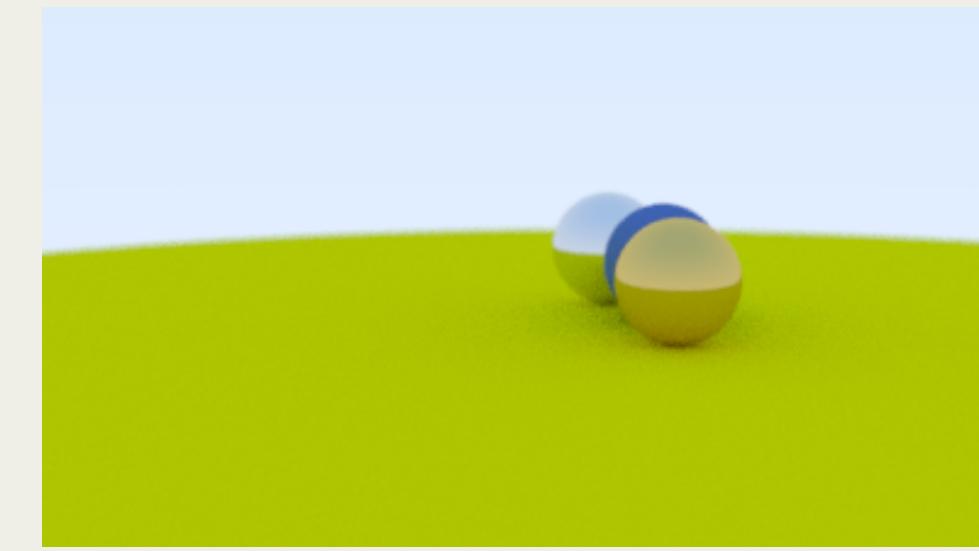
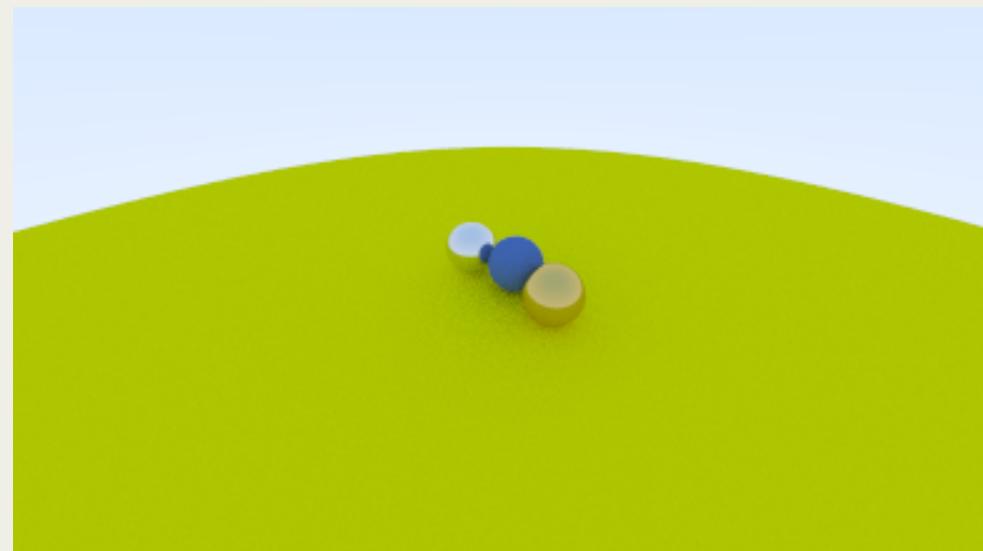
# IMAGES GÉNÉRÉES

---



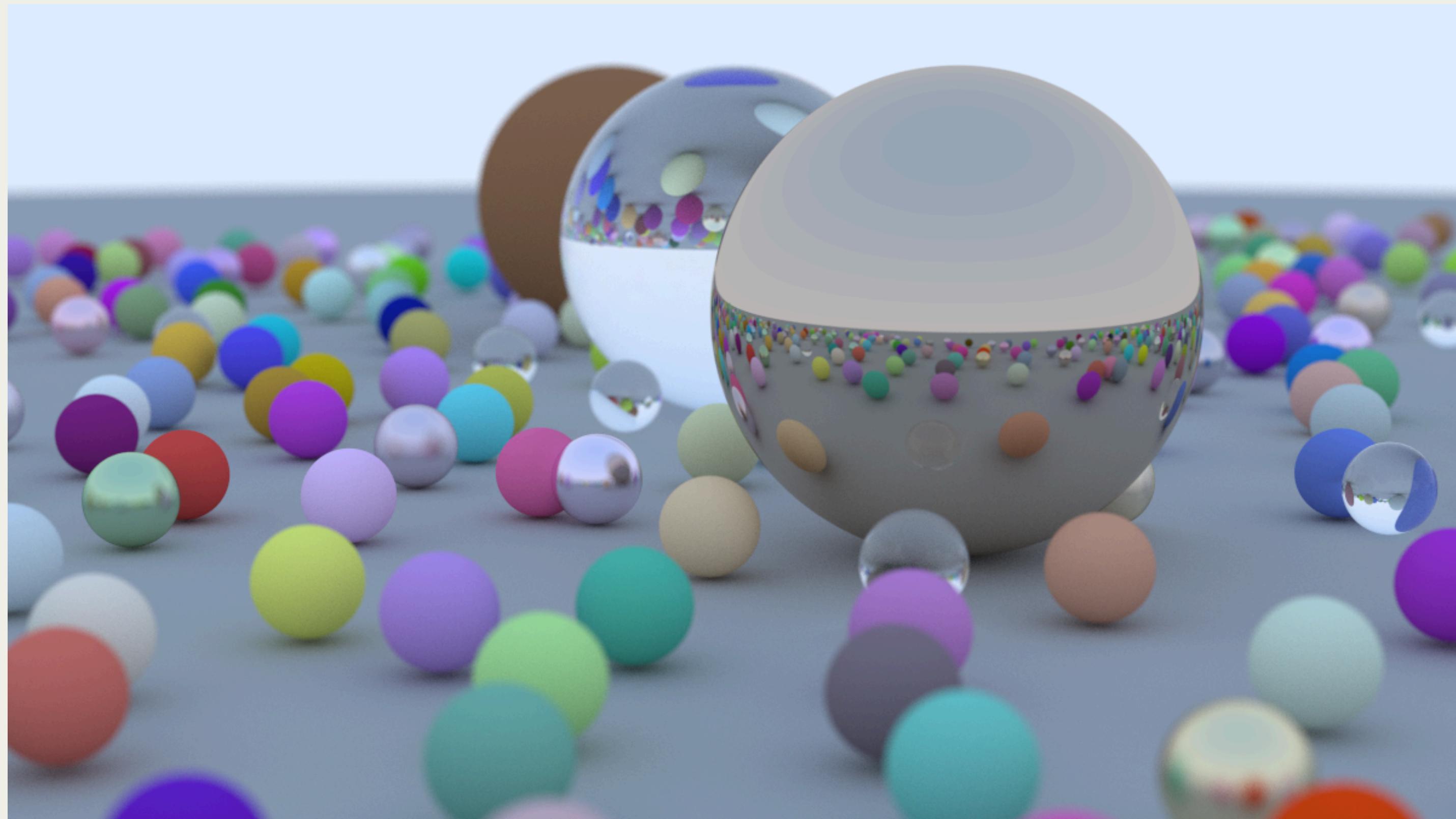
# IMAGES GÉNÉRÉES

---



# IMAGES GÉNÉRÉES

---



# OBJETIF DU PROJET

---

- Concevoir une bibliothèque d'objets pour modéliser les différents composants d'une scène 3D.
- Développer une bibliothèque permettant d'implémenter divers moteurs de rendu basés sur la technique du lancer de rayons (Ray Tracing).
- Créer un environnement permettant :
  - La configuration des scènes,
  - L'exécution des moteurs de rendu,
  - L'affichage des résultats obtenus.

# ARCHITECTURE DU SOFTWARE

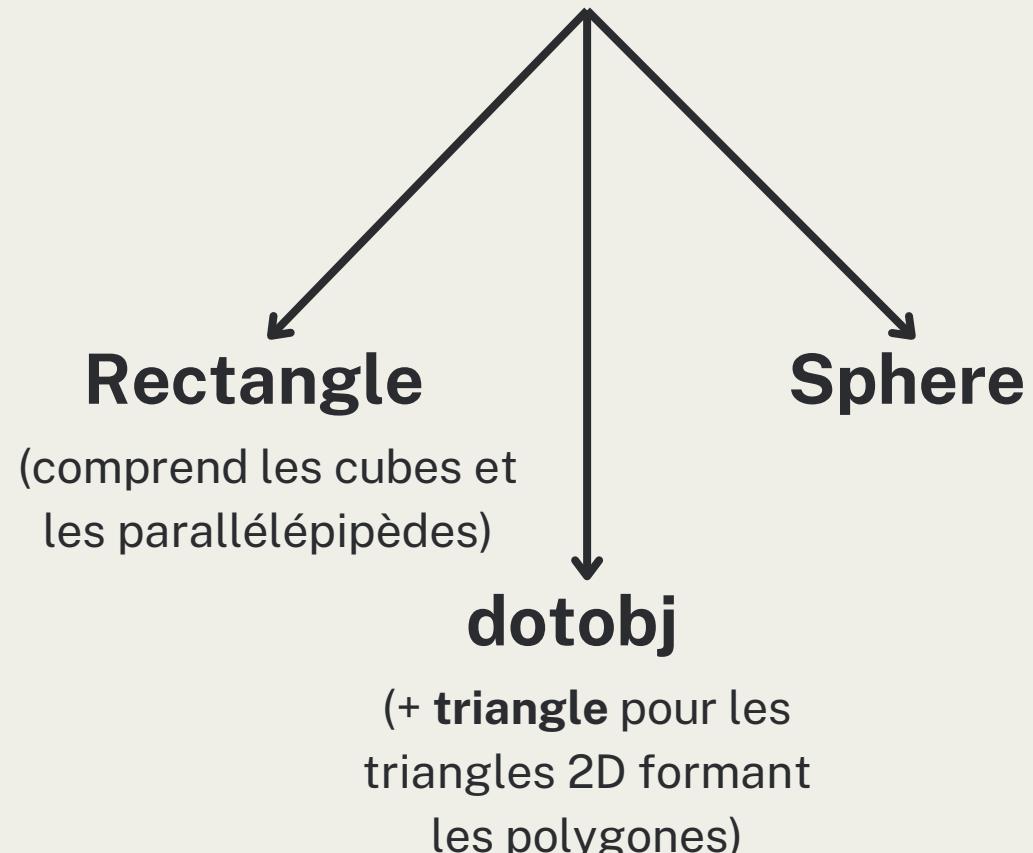
# CLASSES PRINCIPALES DU PROJET

## Camera

Cette classe initialise les paramètres de la caméra (orientation, champ de vision, mise au point). Elle génère des rayons pour chaque pixel de l'image et calcule la couleur correspondante en tenant compte des intersections avec les objets de la scène.

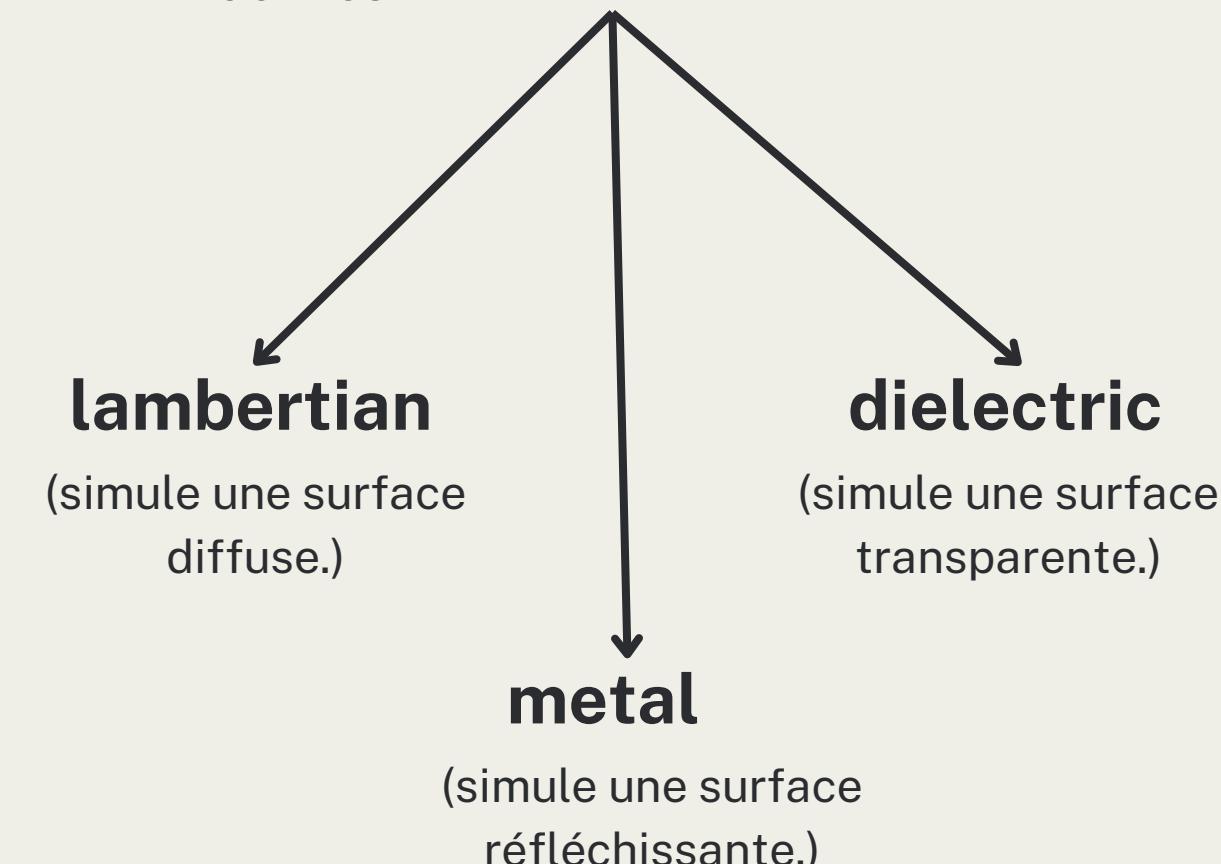
## Object

Classe de base abstraite pour décrire les objets qui seront simulés. Elle permet de définir des méthodes et des propriétés communes pour les objets dérivés.



## Material

Classe de base abstraite pour décrire les matériaux qui seront simulés. Elle permet de définir des méthodes et des propriétés communes pour les matériaux dérivés.



# CLASSES PRINCIPALES DU PROJET

---

## Color

La classe Color représente les couleurs dans l'espace RGB et gère la correction gamma pour un rendu visuel précis.

## Ray

La classe Ray modélise un rayon utilisé pour tracer des trajectoires dans la scène. Elle définit une origine et une direction normalisée, permettant de calculer la position du rayon en fonction d'un paramètre  $t$ .

## RayIntersection

La classe RayIntersection contient les détails et propriétés du point d'intersection entre un rayon et un objet 3D.

# DESCRIPTION DES SCÈNES 3D AVEC DES FICHIERS XML

Les scènes sont décrites à l'aide de fichiers XML, qui contiennent des informations détaillées sur la scène, telles que les positions des objets, la position de la caméra, ainsi que les couleurs et les matériaux des objets. Une classe dédiée lit ces fichiers et extrait les données nécessaires pour les convertir en objets exploitables par le moteur de rendu.

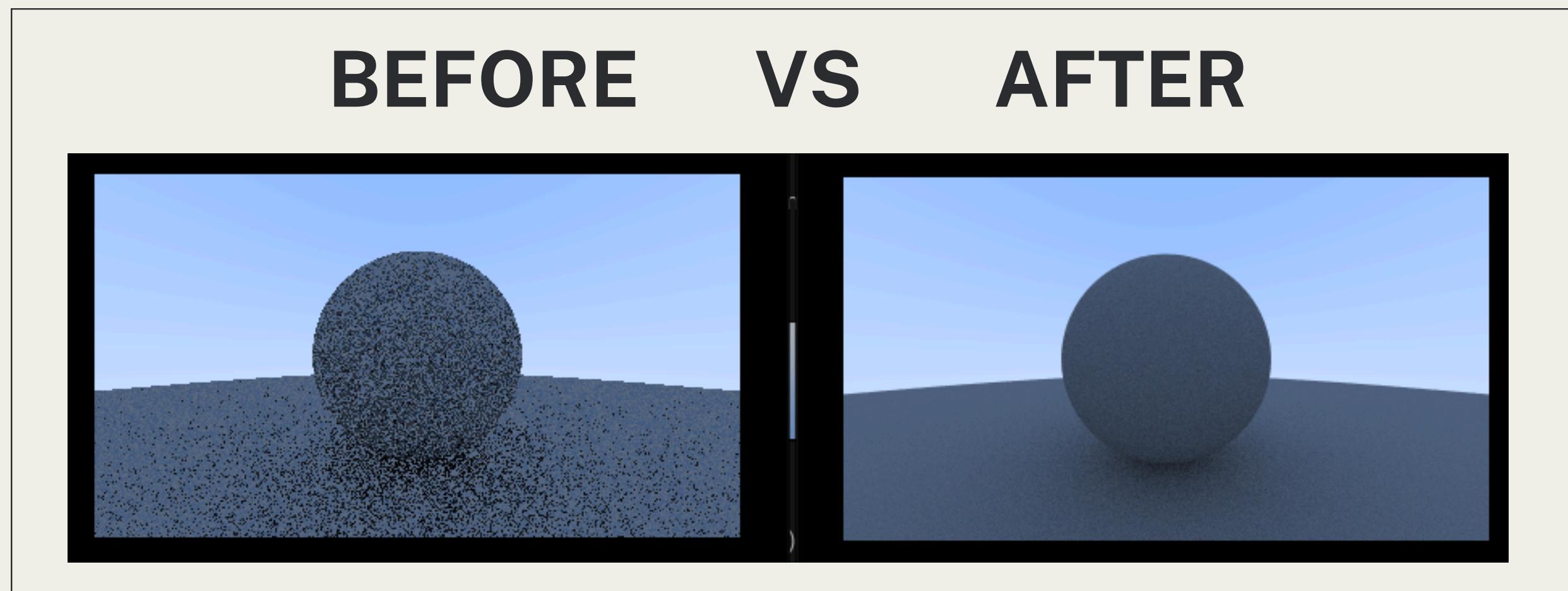
```
1 <scene>
2   <camera>
3     <aspect_ratio>1.7</aspect_ratio>
4     <image_width>800</image_width>
5     <samples_per_pixel>100</samples_per_pixel>
6     <max_depth>50</max_depth>
7     <vfov>10</vfov>
8     <lookfrom>
9       <x>13</x>
10      <y>1</y>
11      <z>3</z>
12    </lookfrom>
13    <lookat>
14      <x>0</x>
15      <y>0</y>
16      <z>0</z>
17    </lookat>
18    <vup>
19      <x>0</x>
20      <y>1</y>
21      <z>0</z>
22    </vup>
23    <defocus_angle>0.1</defocus_angle>
24    <focus_dist>10.0</focus_dist>
25  </camera>
26  <objects>
27    <object type="cube">
28      <position x="-4" y="0" z="0" />
29      <edge>0.5</edge>
30      <material type="dielectric">
31        <refraction_index>0.5</refraction_index>
32      </material>
33    </object>
34    <object type="sphere">
35      <position x="0" y="0" z="0" />
36      <radius>0.25</radius>
37      <material type="lambertian">
38        <color r="1" g="0.5" b="0.5" />
39      </material>
40    </object>
41    <object type="sphere">
42      <position x="-2" y="0" z="0" />
43      <radius>0.5</radius>
44      <material type="metal">
45        <color r="0.5" g="0.5" b="0.5" />
46        <fuzz>0.1</fuzz>
47      </material>
48    </object>
49  </objects>
```

# DIFFICULTÉS RENcontrées

# ALIASING

L'**aliasing** se produit lorsque des lignes ou des textures apparaissent dentelées. Ce problème s'intensifie avec l'introduction du premier modèle de diffusion en raison des interactions non uniformes des rayons avec les surfaces.

L'**anti-aliasing** consiste à lisser ces artefacts visuels en ajustant les couleurs des pixels adjacents pour un rendu plus fluide.



# TEMPS D'EXÉCUTION ÉLEVÉ

Lors des tests de rendu d'image, nous avons remarqué que les temps d'exécution étaient considérablement élevés.

Pour résoudre ce problème, l'optimisation via le **multi-threading** a été mise en place, en envisageant une éventuelle implémentation sur **GPU** avec **CUDA** pour accélérer les calculs et réduire les délais.

before  
VS  
after

```
sinda@sinda-pc:~/9raya/IN204/RayTracer/RayTracer-IN204-latest/RayTracer-IN204-ma
in/build$ ./raytracer
Image générée dans le fichier 'output.ppm'.
Exec time: 10.4366 seconds
sinda@sinda-pc:~/9raya/IN204/RayTracer/RayTracer-IN204-latest/RayTracer-IN204-ma
in/build$ make
[ 7%] Building CXX object CMakeFiles/raytracer.dir/Camera.cpp.o
[ 15%] Linking CXX executable raytracer
[100%] Built target raytracer
sinda@sinda-pc:~/9raya/IN204/RayTracer/RayTracer-IN204-latest/RayTracer-IN204-ma
in/build$ ./raytracer
Image générée dans le fichier 'output.ppm'.
Exec time: 4.41842 seconds
```

# GESTION DE LA MÉMOIRE

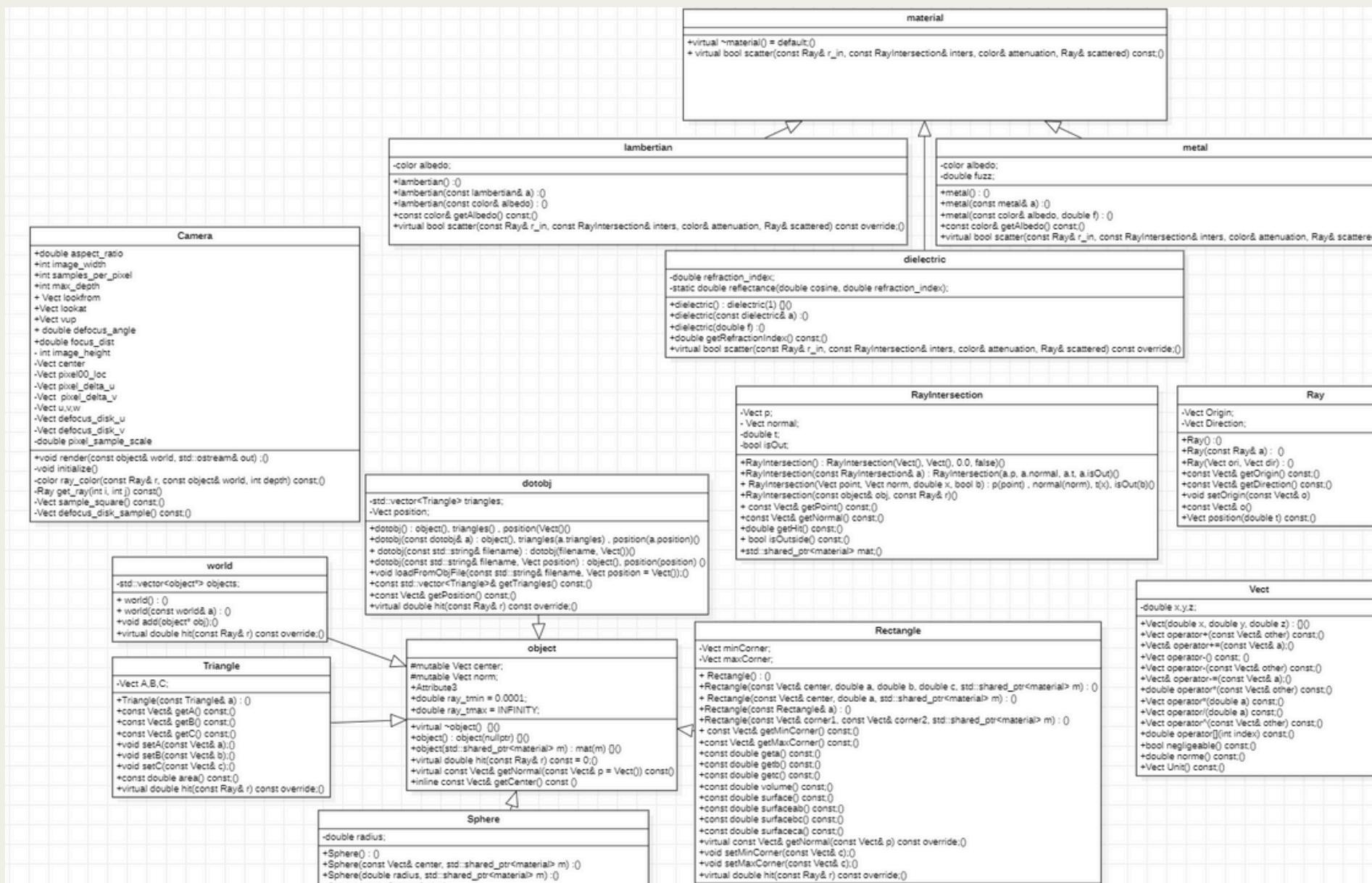
L'optimisation de l'utilisation du tas a été réalisée grâce à l'utilisation de **smart pointers**, comme **std::unique\_ptr** et **std::shared\_ptr**, qui gèrent la mémoire de manière automatique. De plus, l'implémentation de destructeurs dans les classes a été réalisée pour garantir une gestion correcte des ressources, surtout que les threads se partagent certaines de ces ressources.

Le rapport de Valgrind indique que tout ce qui a été alloué a été détruit en fin de tâche

```
--5247-- REDIR: 0x4cb05a0 (libc.so.6:memchr) redirected to 0x483d1c0 (_vgnU_ifunc_wrapper)
--5247-- REDIR: 0x4cb42d0 (libc.so.6:strspn) redirected to 0x483d1c0 (_vgnU_ifunc_wrapper)
--5247-- REDIR: 0x4cb10d0 (libc.so.6:mempcpy) redirected to 0x483d1c0 (_vgnU_ifunc_wrapper)
--5247-- REDIR: 0x4cb3d40 (libc.so.6:strncasecmp) redirected to 0x483d1c0 (_vgnU_ifunc_wrapper)
--5247-- REDIR: 0x4d8c5f0 (libc.so.6:_strrchr_avx2) redirected to 0x484ed20 (rindex)
--5247-- REDIR: 0x4d8a740 (libc.so.6:_strlen_avx2) redirected to 0x484f220 (strlen)
--5247-- REDIR: 0x4d872a0 (libc.so.6:_memcmp_avx2_movbe) redirected to 0x4852480 (bcmpl)
--5247-- REDIR: 0x4d8b820 (libc.so.6:_strncmp_avx2) redirected to 0x484fab0 (strncmp)
--5247-- REDIR: 0x4cac640 (libc.so.6:malloc) redirected to 0x48467b0 (malloc)
--5247-- REDIR: 0x4d89b60 (libc.so.6:_strchr_avx2) redirected to 0x484ef00 (index)
--5247-- REDIR: 0x4925950 (libstdc++.so.6:operator new[](unsigned long)) redirected to 0x4848550 (operator new[](unsigned long))
--5247-- REDIR: 0x49258e0 (libstdc++.so.6:operator new(unsigned long)) redirected to 0x484ef30 (operator new(unsigned long))
--5247-- REDIR: 0x4d87a00 (libc.so.6:_memcpy_avx_unaligned_erms) redirected to 0x4852d60 (memmove)
--5247-- REDIR: 0x49238a0 (libstdc++.so.6:operator delete(void*)) redirected to 0x484a080 (operator delete(void*))
--5247-- REDIR: 0x49238d0 (libstdc++.so.6:operator delete[](void*)) redirected to 0x484bec0 (operator delete[](void*))
--5247-- REDIR: 0x4cacd20 (libc.so.6:free) redirected to 0x4849820 (free)
==5247==
==5247== HEAP SUMMARY:
==5247==     in use at exit: 0 bytes in 0 blocks
==5247== total heap usage: 993 allocs, 993 frees, 157,176 bytes allocated
==5247==
==5247== All heap blocks were freed -- no leaks are possible
==5247==
==5247== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

# ARCHITECTURE ÉVOLUTIVE

Le projet devrait relever le défi d'être flexible et évolutif en utilisant le polymorphisme et des classes abstraites, ce qui permettra d'ajouter facilement de nouvelles fonctionnalités ou de modifier le comportement des éléments existants sans perturber le reste du système.



**Extrait du  
diagramme UML du  
projet.**

# AMÉLIORATION ENVISAGÉE

- Intégration du support des textures complexes, notamment pour la classe .obj, qui actuellement se limite à une seule couleur prédéfinie au lieu de prendre en charge les textures générées par Blender.



- Ajout de la prise en charge des sources lumineuses mobiles pour une gestion dynamique de l'éclairage.



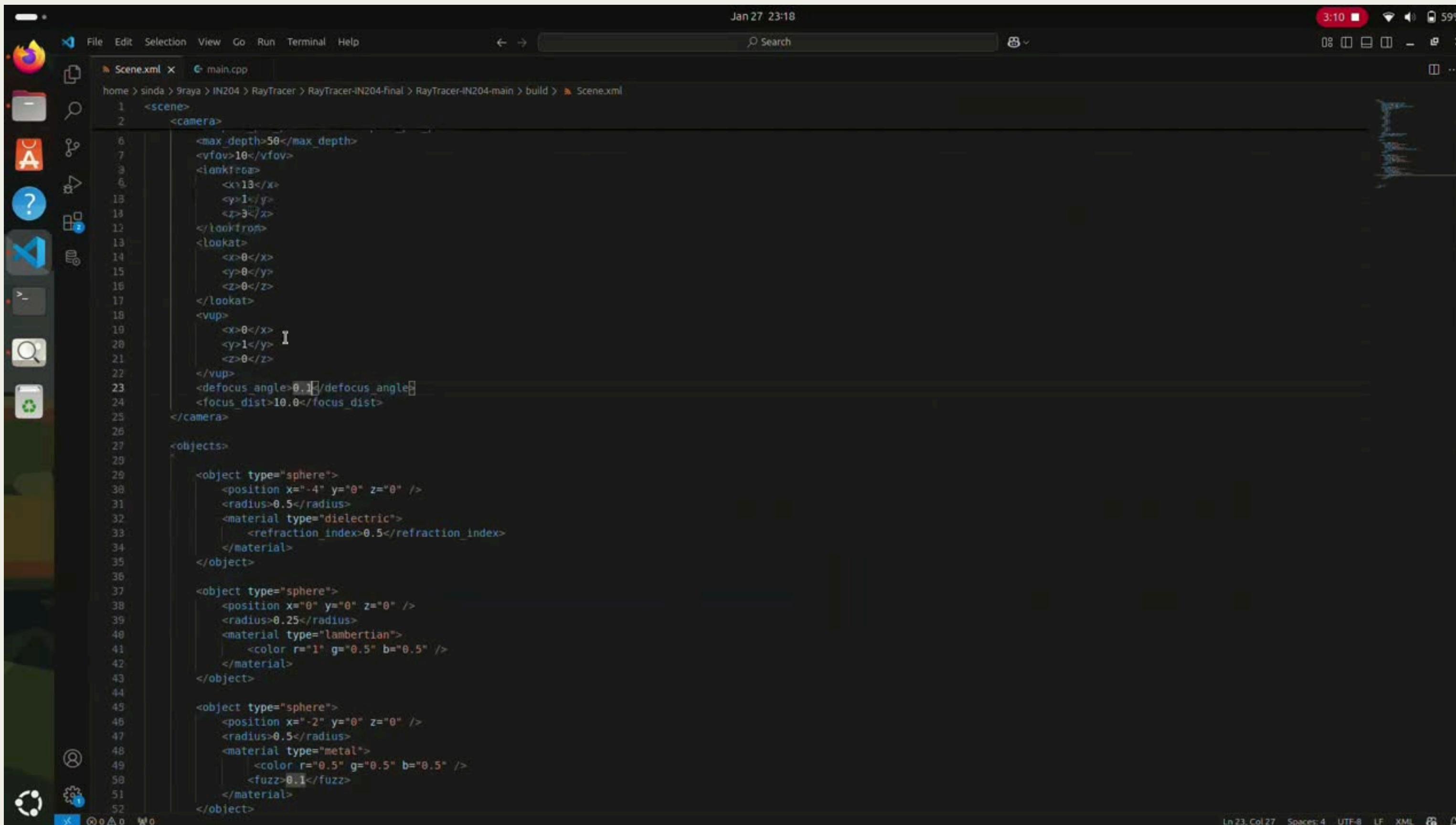
- Amélioration des performances grâce à l'utilisation du GPU pour les calculs intensifs.



- Développement d'une interface graphique conviviale pour simplifier la description et la configuration des scènes.



# DÉMONSTRATION



A screenshot of a terminal window showing the command "raytracer -r 1000x500 -o output.png" being run. The terminal has a dark theme with white text on a black background. The command is entered in the first line, and the output shows the progress of the rendering process, indicating it's at 100% completion.

```
RayTracer -r 1000x500 -o output.png
[...]
[100%]
```

# Merci pour votre attention!

---

 github : <https://github.com/MedMzi/RayTracer-IN204>

