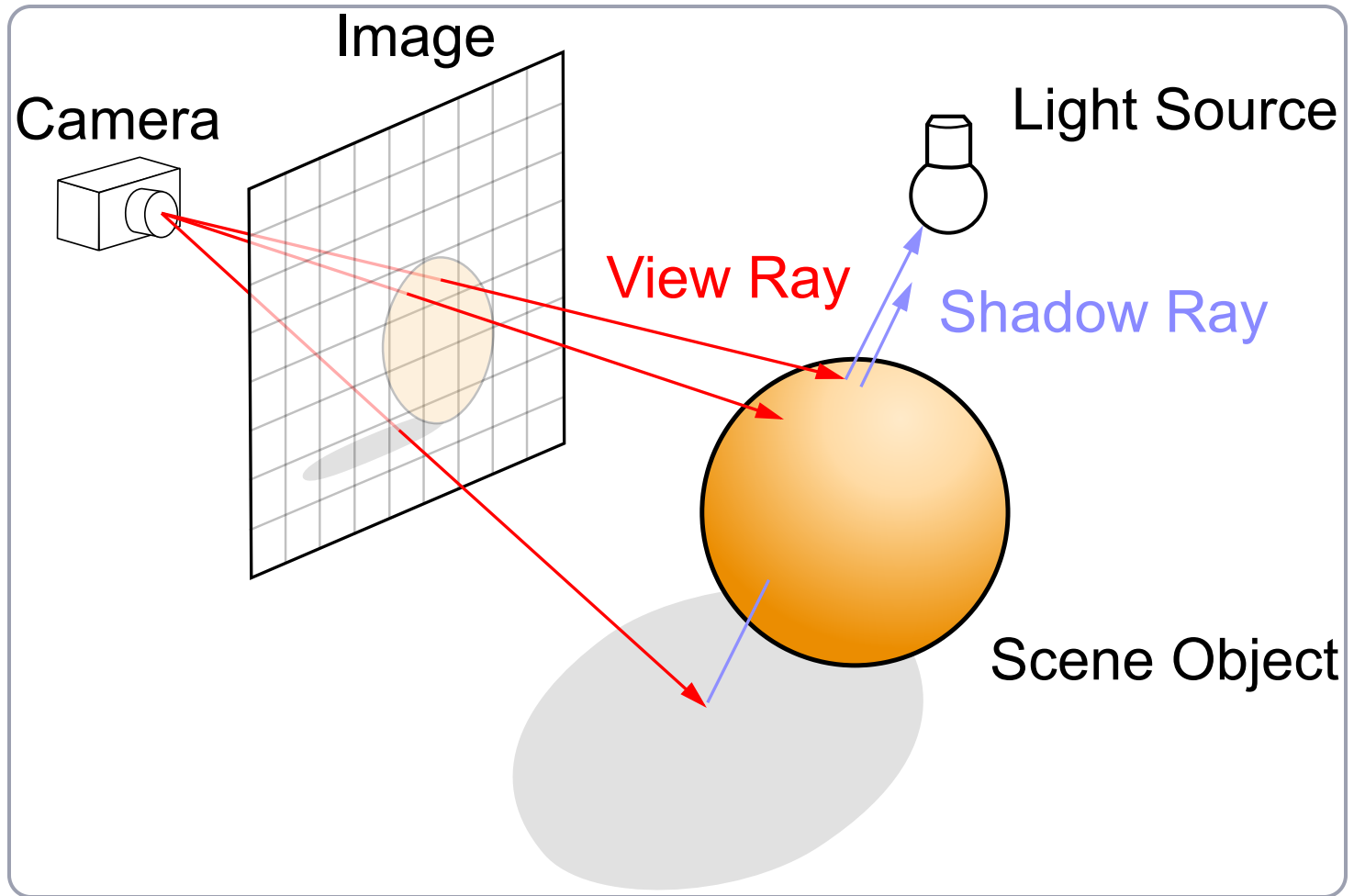


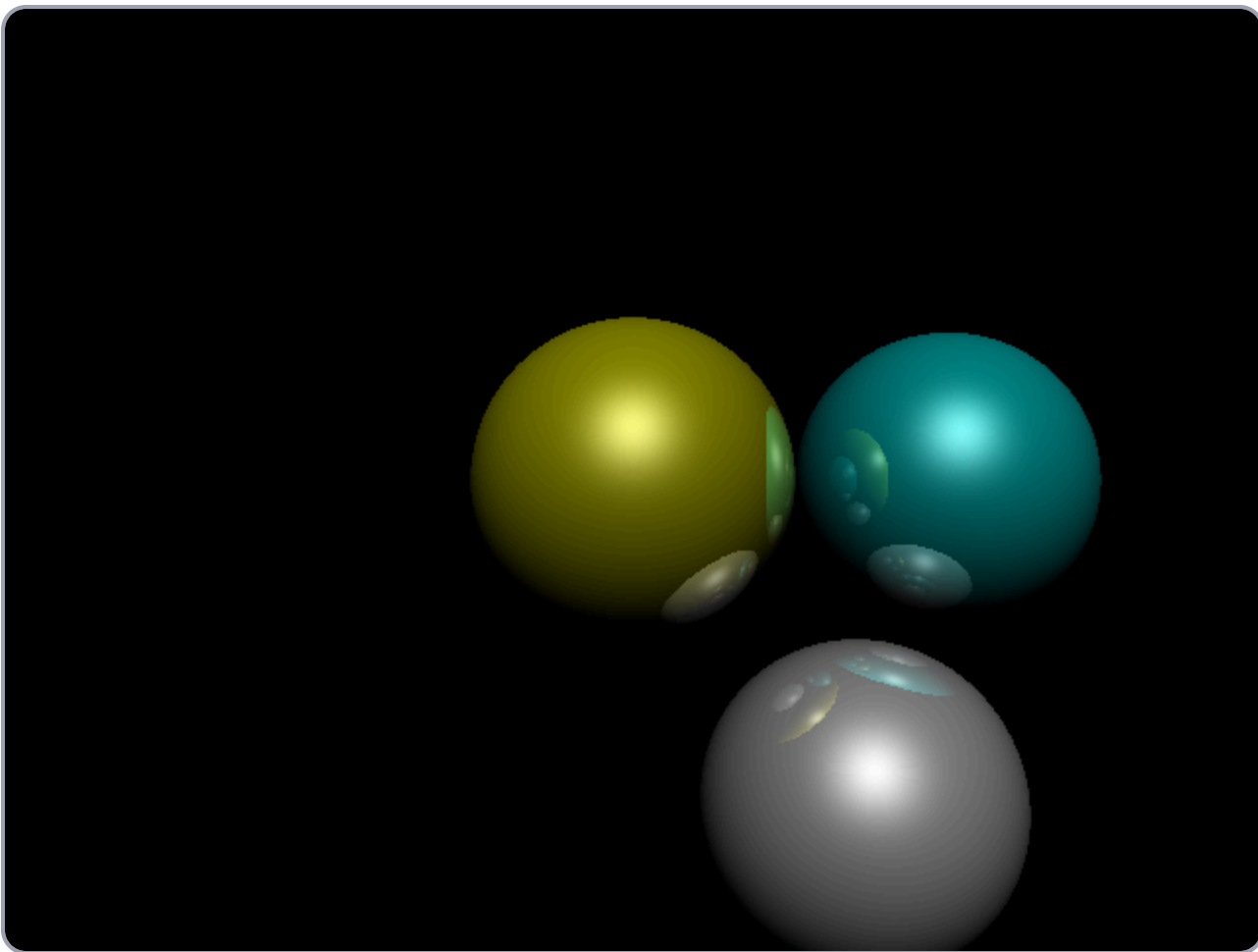
Projet Ray Tracer - Sujet

Présentation

Le but de ce projet est de réaliser un [lanceur de rayons](#) basique.



L'objectif sera d'obtenir ce genre de rendus :



Le projet est découpé en **jalons**, chaque jalon permet d'ajouter une ou plusieurs fonctionnalités à votre RayTracer. L'objectif est d'avancer à raison d'**un jalon par semaine** (au moins), et de poser toutes les questions dont vous aurez besoin lors des sessions en distanciel.

Le projet sera réalisé en **Java** (une version récente de préférence)

Nous fournirons des fichiers de description de scène, et des **exemples de rendu** pour chacun de ces fichiers, ce qui permettra de tester votre programme et de vérifier qu'il fait bien ce qu'il est censé faire.

Pour réaliser ce Ray Tracer, vous serez amenés à manipuler les concepts suivants :

- Programmation Orientée Objet
- Parsing de fichiers
- Écriture d'images
- Utilisation de bibliothèques standard en Java

Remarque : les personnes sachant utiliser git sont vivement encouragées à l'utiliser pour ce projet. L'IMT met un Gitlab à disposition, utilisez-le pour sauvegarder votre travail régulièrement.

Pour les autres, vous pouvez au choix :

- Apprendre à utiliser git par vous même et l'utiliser
- Faire des archives de **sauvegarde régulières** de votre projet (à chaque jalon au moins)

Jalons

Fonctionnement des jalons

Chaque jalon est conçu de la manière suivante :

- L'**objectif** du jalon est décrit
- Une **spécification** fonctionnelle est proposée
- Si nécessaire, un paragraphe donne des **explications théoriques** plus détaillées pour comprendre ce que vous implémentez.
 - **Toutes les équations mathématiques sont fournies, aucun calcul supplémentaire n'est nécessaire**
 - Nous explicitons quelles formules sont à utiliser
 - L'objectif est de vous faire travailler la programmation, pas les mathématiques
- Pour finir, une liste d'**aides à la conception** est proposée.
 - Chaque indice donne plus d'informations pour vous guider
 - Essayez de ne pas les utiliser dans un premier temps, cherchez par vous-même
 - Puis lorsque vous êtes bloqué.e, regardez l'indice suivant et essayez de vous débloquer
 - Les aides seront de moins en moins détaillées au fil des jalons

Les **dates** proposées sont indicatives, il n'y a pas de rendu intermédiaire prévu. Mais pour plus d'efficacité lors des séances à distance, essayez de terminer le jalon prévu. Cela nous permettra de vous apporter une aide pertinente et régulière.

Essayez de garder un **rythme** régulier : le projet est long, et il sera impossible de le faire en quelques heures la veille du rendu final.

Le jalon 6 correspond à la date de rendu finale. Il ne propose que des étapes bonus pour les personnes ayant réussi le reste du projet. Certaines étapes seront faisables avant d'avoir fini le jalon 5 et nous vous l'indiquerons dans les jalons concernés.

Pour les autres personnes, la dernière semaine (entre les jalons 5 et 6) sera consacrée à la finalisation du projet et les derniers correctifs.

Liste des jalons

- Jalon 0 - 31/10/2025 - Comparateur d'images (TD 3)
- Jalon 1 - 04/11/2025 - Classes de bases pour le calcul vectoriel
- Jalon 2 - 12/11/2025 - Lecture du fichier de description de scène
- Jalon 3 - 18/11/2025 - Premières images (sphères uniquement)
- Jalon 4 - 25/11/2025 - Calcul de la couleur d'un point
- Jalon 5 - 02/12/2025 - Ajout des ombres et de l'illumination de Phong
- Jalon 6 - 09/12/2025 - **(Rendu)** Finalisation et bonus

- Ajout des plans et des triangles
- Surfaces réfléchissantes

Rendu

- Archive ZIP contenant
 - le code final du projet.
 - Le code doit compiler sans nécessiter de dépendances supplémentaires autres que JUnit (pour les tests uniquement) : Java vanilla uniquement.
 - Fournissez un exécutable compilé et fonctionnel.
 - Code documenté (JDoc)
 - Tests unitaires et d'intégration présents (pas nécessaire d'avoir 100% de coverage)
 - Obligatoires sur : calculs vectoriels
 - Un rendu PNG du fichier de description `final.scene` obtenu avec votre code.