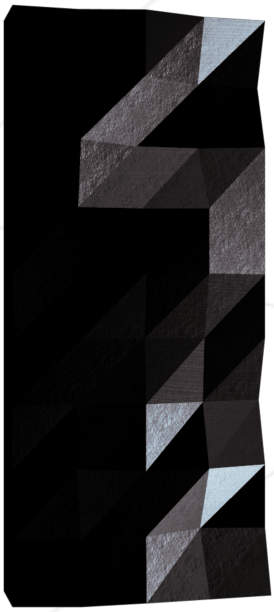

GROS Projet d'Infographie

RT

Résumé: Ce projet n'est rien d'autre qu'une apothéose d'images de synthèse fabuleuses.

Table des matières

Chapitre I

Préambule

Les pantoufles chantaient dans l'azur, faméliques,
aux rythmes entrelacés de mâles abricots,
tandis que les homards volant aux alentour des portiques
pour y mieux voir, otaient leurs tricots.

C'est alors que surgit mon oncle d'amérique
l'oeil vélocipédique et flanqué d'asticots
portant sur un plateau d'or des moutons arthritiques,
un vieux mouchoir rempli de vieux os et chicos.

Montant sur son impérial,
les baromètres en deuil ont contracté la gale
passant brusquement de la vie au trépas.

Le pétrôle, assis au milieu de la route,
contemplait d'un oeil morne et ne comprenait pas.

Oncle Potam
d'apres Roger Lefebvre (1916-2005)

Chapitre II

Sujet

II.1 Les objectifs

Votre objectif est d'être capable, à l'aide de votre programme, de générer des images de synthèse selon la méthode du Ray-Tracing. Ces images de synthèse représentent une scène, vue d'une position et d'un angle spécifiques, définie par des objets géométriques simples, entiers ou partiels, et disposant d'un système d'éclairage.

Le projet comporte une partie obligatoire et de nombreuses options. La partie obligatoire vaut 0 points et les options ne rapportent des points que si la partie obligatoire est complète à 100%. Le projet ne sera validé que si un volume substanciel d'options est présent lors de la soutenance.

II.2 La partie obligatoire

Les éléments à réaliser sont les suivants :

- Coder en C, C++ ou rust.
- Avoir un Makefile normal (tout ce que vous avez l'habitude d'y mettre)
- Implémenter la méthode du lancer de rayon (le raytracing quoi..) pour obtenir une image de synthèse
- Avoir au moins 4 objets géométriques simples comme objets de base (non composés) : plan, shpère, cylindre et cône
- Gestion du réaffichage sans re-calcul (en gros avec la MinilibX on gère l'expose correctement) : si une partie de la fenêtre doit etre redessinée à l'écran, c'est mieux si il n'y a pas tous les calculs à refaire.. sauf si vous faites un RT temps réel. Là vous me mailez et on en discute.
- Position et direction quelconque du point de vision, et des objets simples
- Gestion de la lumière : multi-spot, luminosité, brillance, ombres

Ces éléments sont ceux demandés dans l'ancien RTv1. Pour le RT, vous n'aurez pas le moindre point (et donc aucune chance de valider le projet) si vous ne faîtes que ça.

II.3 Les options illimitées

Passons maintenant au plat de résistance : les options.
Elles sont nombreuses et sans réelles limites. On pourra citer par exemple :

- Lumière d'ambiance
- Objets limités : parallélogrammes, disques, demi-sphère, tuyau, ...
- Perturbation de la normale et de la couleur
- Fichier externe de description de la scène
- Lumière directe
- Lumière parallèle
- Réflexion
- Transparence
- Modification de l'ombre selon la transparence des objets
- Objets composés : cube, pyramide, tétraèdre, ...
- Textures
- Objets négatifs
- Perturbation de la limite / transparence / réflexion, selon une texture
- Autres objets natifs : paraboloïde, hyperboloïde, nappes, tores, ...
- ...

Ça ce sont les trucs à peu près normaux. Bien sûr vous pouvez faire beaucoup plus exotique ! On peut penser par exemple à du calcul distribué sur plusieurs machines, ou encore utiliser des quadriques ou des quartiques, des séquences vidéos réalisées à partir de vos images, une version stéréoscopique pour Oculus Rift, ... Quelques remarques cependant : pour ceux qui veulent s'amuser à parser des fichier .pov ou .3ds, il faut que votre Raytracer soit tout de même capable de gérer correctement les objets simples de la partie obligatoire à partir des équations, pas avec des facettes. De plus, la soutenance demandera des manipulations en live de votre scène. Cela devra se faire avec vos outils et non avec 3DS (bref que vous ayez votre propre fichier de conf en gros).

Un bonus spécial sera attribué au premier groupe qui réalise le raytracing d'un ruban de Moebius (ou Möbius comme vous voulez) à partir de l'équation mathématique, pas en facette.

Vous trouverez dans l'e-learning des exemples techniques pour illustrer certaines options.

Pour la soutenance, il serait plus que souhaitable que vous ayez au moins les 3 scènes suivantes, facilitant le contrôle des éléments obligatoires à faire :

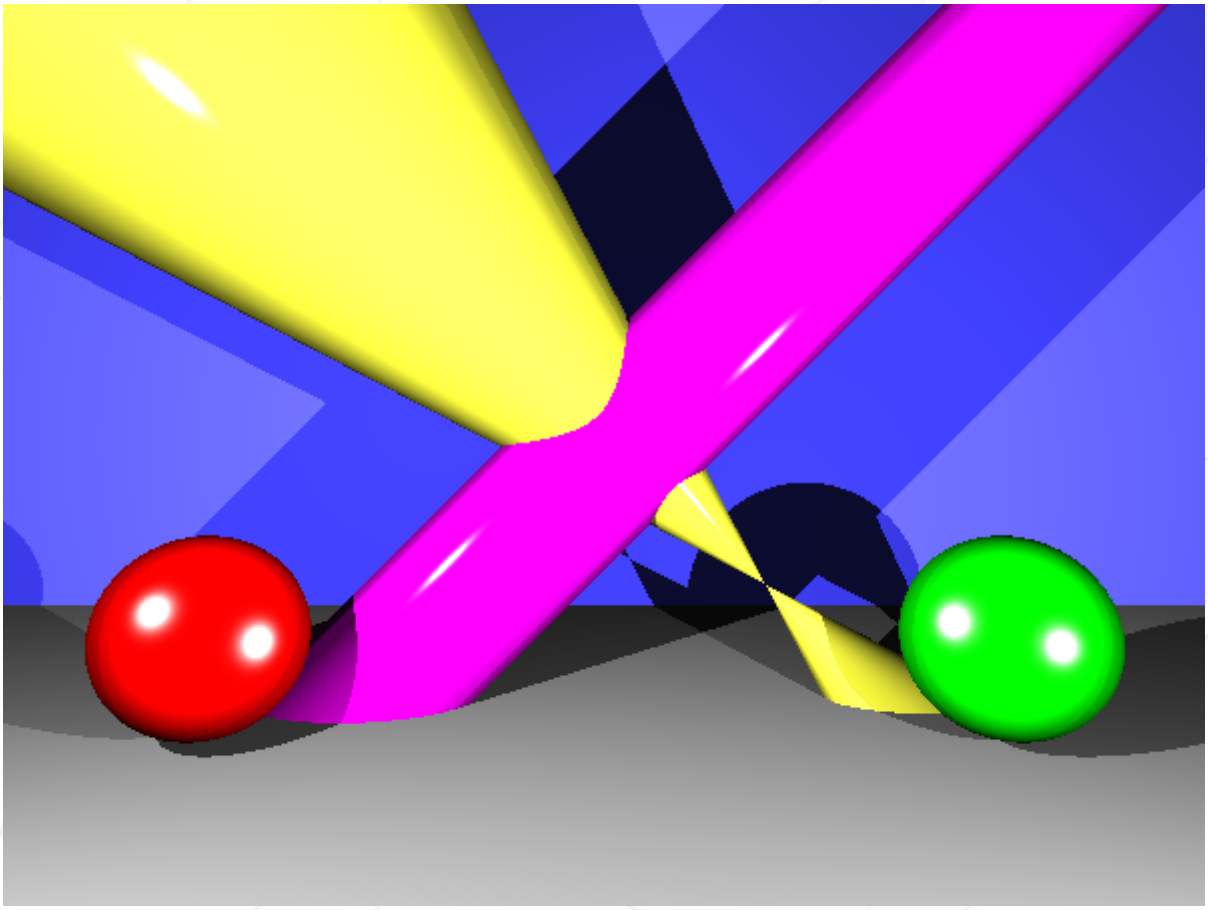

FIGURE II.1 – Les 4 objets, 2 spots, ombres et brillance

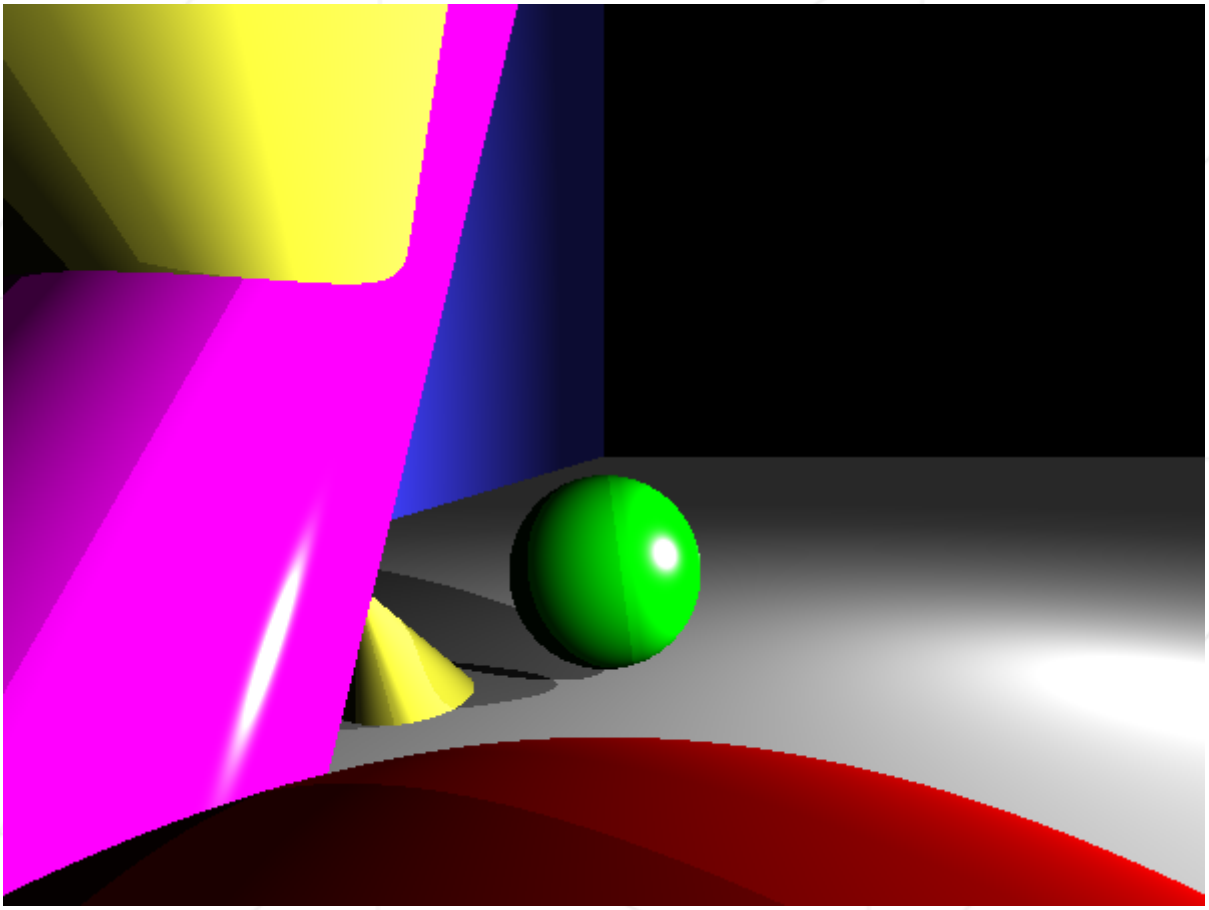

FIGURE II.2 – Même scène, autre point de vision

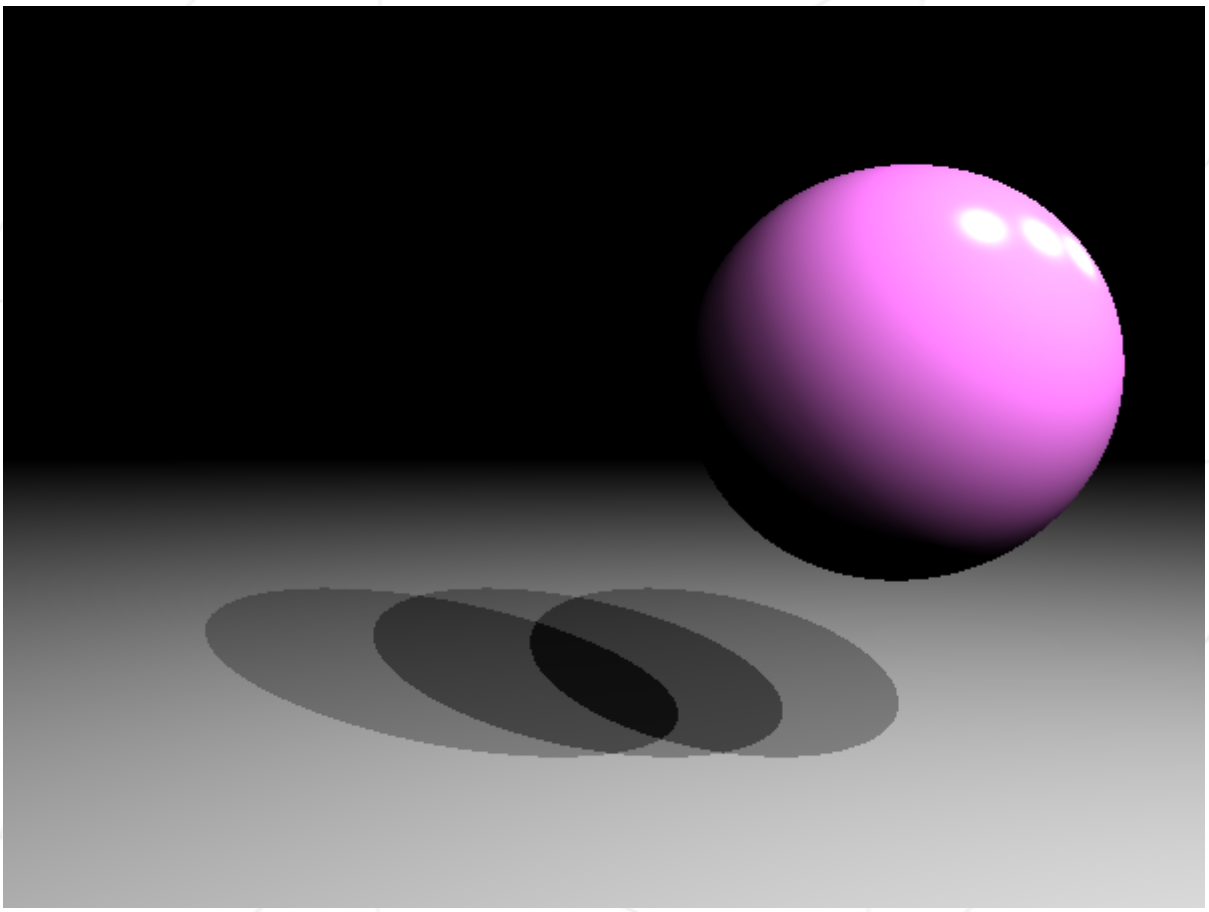

FIGURE II.3 – Viens faire toi-même le mélange des ombres, ... (folklore toulousain)

II.4 Détails

Rendu classique sur le git, Makefile classique, et bien sûr, seul le contenu de votre dépot sera utilisé en soutenance.

Vous devez utiliser en priorité les fonctions qui doivent être présentes (et qui normalement le sont) dans votre libft, en lieu et place de celles de la libc. Vous ne pouvez pas utiliser des librairies non présentes par défaut sur le dump de l'école, à l'exception de votre libft et de la minilibX native MacOS. Un conseil : restez simple, libft, libm et minilibx suffisent.

Vous pouvez utiliser les types float et/ou double et/ou long double du C.

Et si vous déprimez en lisant ce sujet, pensez au chemin parcouru par les fadas de chez ILM, Pixar (ou tout autre bon studio d'image de synthèse qui a déployé des trésors d'ingénierie pour parfaire son art), et détendez-vous en prenant la vie du bon coté : http://www.youtube.com/watch?v=0NgKeRRKE5o (j'ai une préférence pour la VO, mais on la trouve moins facilement en bonne qualité..).