MirrorVerse

Élèves : Guillaume Calderon, Mohammed Ali, Eymeric Déchelette

Enseignant : Jérôme Bastien

Cahier des charges

Contexte

Ce projet prend la suite d'une demande d'un élève, Quentin COURDERO en troisième année en informatique à Polytech. Il a demandé à son enseignant Jérôme Bastien de l'aider à écrire un algorithme pour déterminer le trajet d'un rayon lumineux qui vient frapper (ou pas) un miroir plan fini.

Objectif

L'objectif de ce projet est d'étudier le comportement d'un rayon lumineux lorsqu'il rencontre des miroirs.

Un rayon lumineux peut avoir deux comportements : il peut se retrouver piégé dans le nid de mirroirs, ou parvenir à sortir du nid de mirroirs.

Sa trajectoire peut suivre un motif ou être chaotique.
On considérera qu'une trajectoire est chaotique si, après n réflexions (n dépendant du cas étudié), on ne constate aucune répétition.

Pour ce faire, on codera un outil simulant le comportement de rayons lumineux lorsqu'ils rencontrent des miroirs.
La simulation devra physiquement être juste, c'est à dire coller au maximum à la réalité.
Elle s'appuiera sur la seconde loi de Snell-Descartes et devra obligatoirement fonctionner en 2 dimensions avec des miroirs plans.
Elle pourra être enrichie avec plus de dimensions et de variété de miroirs.

Réponse technique

Le simulateur sera développé avec le langage Rust afin d'avoir un maximum d'optimisation et de s'assurer d'un minimum de bugs imprévus.

On utilisera la librairie nAlgebra afin de pouvoir manipuler aisément différentes notions mathématiques telles que les vecteurs, les points, etc.

La simulation disposera d'un outil de visualisation permettant de se déplacer dans le monde virtuel pour constater simplement le résultat de la simulation.

L'outil de visualisation sera développer à l'aide de la librairie wgpu.

Potentiel difficulté

La première difficulté sera de détecter efficacement l'intersection entre les rayons et les miroirs.

La technologie d'affichage demandera aussi probablement beaucoup de recherche documentaire en raison du fait qu'elle est assez nouvelle pour les réalisateurs du projet.

Milestones

Fonctionnalités v1

- On devra pouvoir éditer facilement l'ensemble des miroirs pour la simulation. Probablement via une simple description en JSON.
- On devra également pouvoir choisir la direction et le point de départ du rayon.
- On devra pouvoir visualiser aisément le trajet du rayon lumineux.
- On devra supporter les miroirs plans.
- La simulation devra au moins fonctionner en 2D.
  La V1 utilisera des bases locales et des symétries plutôt que des angles afin d'anticiper la généralisation en 3D.

Fonctionnalités v2

- On devra supporter les types de miroirs :
  - plan
  - circulaire
  - en courbe de Bézier

Fonctionnalités v3

- La simulation devra au moins fonctionner en 3D (ou ND).

Fonctionnalités v4

- on devra ajouter, selon les besoins, des fonctionnalités d'analyse de la trajectoire du rayon :

- détection automatique de la sortie de l'ensemble.
- détection automatique d'une boucle (le rayon passe 2 fois au même endroit)

Organisation temporelle