Centrale Informatique MP-PC-PSI 2021 — Corrigé

Ce corrigé est proposé par Cyril Ravat (professeur en CPGE) ; il a été relu par Julien Dumont (professeur en CPGE) et Benjamin Monmege (enseignant-chercheur à l'université).

Ce sujet a pour objectif la réalisation d'une image en deux dimensions représentant une scène simple, composée de plusieurs sources lumineuses et de plusieurs objets, choisis sphériques pour simplifier le problème. Il permet d'aborder une technique très efficace de fabrication d'images virtuelles et demandant peu de ressources. Les structures manipulées sont des tableaux `numpy` et des fonctions sont données pour en faciliter l'usage.

- La partie I permet la construction des fonctions élémentaires de géométrie et des premiers objets optiques tels que rayons, sphères et intersections des deux.
- Dans la partie II, l'étude évolue vers la problématique de la visibilité d'une source lumineuse depuis un point d'une sphère et de la couleur obtenue compte tenu de la lumière de la source et de l'incidence du rayon lumineux.
- La partie III, indépendante du reste de l'énoncé, demande de traiter l'enregistrement des données relatives à une scène (positions et couleurs des objets et des sources) au sein d'une base de données. C'est l'occasion de poser quatre questions de langage SQL d'un niveau très progressif.
- L'étude entre dans le vif du sujet à la quatrième partie, avec la modélisation du « lancer de rayons », considération physiquement fausse mais compatible avec la réalité qui consiste à imaginer que les rayons lumineux partent de l'œil. On modélise ces rayons et leur première intersection avec les objets de la scène, puis on étudie la couleur vue et la projection de cette même scène sur un écran en deux dimensions.
- La partie V, plus difficile que les précédentes, demande de ne pas se perdre dans les notations et le concept de rayon lumineux se réfléchissant d'une surface à l'autre. Les deux dernières questions portent sur une optimisation qui reste quelque peu théorique par rapport au reste du sujet.

Ce sujet donne une bonne idée de ce que l'on peut faire avec des outils informatiques, sur le thème assez pratique de la réalisation d'images de synthèse. Il est globalement bien mené et progressif, tous les concepts sont parfaitement définis. Il ne contient que de la manipulation assez simple de tableaux `numpy`, accessible dès la première année. Il a le grand intérêt de faire produire quasiment l'ensemble du code nécessaire à la mise en œuvre : le lecteur intéressé pourra rapidement s'amuser à créer lui-même des images. La fin du sujet réserve des questions plus délicates pour les candidats plus à l'aise.

Indications

Partie I

1 Bien lire les paragraphes précédant la question, notamment à propos de la représentation des points et des vecteurs.

2 Lire attentivement l'annexe et les fonctions du module `numpy` autorisées.

5 Le mot-clé `assert` permet la vérification de la condition donnée et arrête l'exécution si elle n'est pas vérifiée.

8 Il faut résoudre l'équation précédente. On peut remarquer que les deux solutions possibles doivent obligatoirement être positives.

Partie II

10 Faire un dessin en deux dimensions et regarder les vecteurs présents.

11 Il faut chercher les intersections du rayon partant de la source lumineuse et passant par P. Ne pas oublier que les sphères plus éloignées de la source que P n'interviennent pas.

12 Le code à produire est très court, il suffit de traduire la relation (2) de l'énoncé. Le cosinus de θ peut être obtenu d'après le schéma par un produit scalaire.

13 Faire un schéma en deux dimensions et décomposer $\overrightarrow{u}$ sur l'axe de $\overrightarrow{\mathrm{N}}$ et l'axe orthogonal.

Partie III

14 Pour récupérer l'année, utiliser une fonction donnée en annexe.

16 Regarder où se trouvent les données permettant la sélection et celles nécessaires à la projection, puis déterminer comment joindre les tables concernées.

17 La fonction `OCCULTE` demande deux identifiants d'objets indépendants, il faut donc joindre deux fois la table `Objet` en les renommant.

Partie IV

18 Déterminer le lien entre x et j, puis entre y et i. Pour cela, faire un schéma représentant les cases et le centre de la figure, pour une valeur de N faible. Attention, sur la figure de l'énoncé i et j ne sont pas en face de $\mathrm{E}(i, j)$.

20 Pour retourner le point d'intersection le plus proche et l'indice de l'objet correspondant, il est nécessaire de récupérer toutes les intersections avec la distance à l'œil, le point de contact et l'indice de l'objet afin de conserver les bonnes valeurs.

21 Le calcul de couleur diffusée depuis une source par un point de sphère a déjà été fait à la question 12. Il suffit d'ajouter les couleurs correspondant aux différentes sources, sans oublier de vérifier si elles sont bien visibles.

Partie V

25 Une boucle `while` ou `for` est possible. À chaque itération, si l'interception du rayon existe, on la stocke et on continue.

26 La relation donnée dans l'énoncé implique que l'on doit parcourir le tableau des réflexions en partant de la fin.

27 Reprendre la fonction `lancer` et regarder ce que `couleur_perçue` y modifie.

29 Créer l'ensemble des triplets {`objr_id`, `so_id`, `objo_id`}. Utiliser l'opérateur `in` pour vérifier leur présence dans la liste `risque`.

30 Reprendre la fonction `visible` et l'adapter. Attention à l'ordre des arguments.

I. Géométrie

1 La fonction `vec` calcule les composantes du vecteur $\overrightarrow{AB}$ en soustrayant les coordonnées du point A à celles du point B.

```
def vec(A, B):
    return B - A
```

> L'énoncé dit explicitement que retrancher deux vecteurs est réalisé par l'opération `v1 - v2` et que les vecteurs, comme les points, sont représentés par des tableaux. Il fallait donc éviter d'écrire une boucle ici.

2 La fonction `ps` calcule le produit scalaire de deux vecteurs, somme des produits terme à terme de leurs composantes, ce que fait précisément la fonction `np.inner`.

```
def ps(v1, v2):
    return np.inner(v1, v2)
```

> La méthode `inner` du module `numpy` est donnée dans l'annexe et sa description correspond exactement au produit scalaire. Il est néanmoins possible de réaliser cette opération autrement, notamment en faisant la somme des composantes du produit de Hadamard des deux vecteurs :
>
> ```
> def ps(v1, v2):
>     return np.sum(v1 * v2)
> ```
>
> La réalisation de ce calcul par une boucle, demandant beaucoup plus d'écriture, a certainement été acceptée par les correcteurs, mais ce n'est visiblement pas la réponse attendue. On peut écrire par exemple
>
> ```
> def ps(v1, v2):
>     somme = 0
>     for i in range(len(v1)):
>         somme += v1[i] * v2[i]
>     return somme
> ```

3 La fonction `norme` calcule la racine carrée du produit scalaire du vecteur par lui-même.

```
def norme(v):
    return math.sqrt(ps(v, v))
```

4 La fonction `unitaire` retourne le vecteur unitaire correspondant à l'argument, en utilisant le produit d'un scalaire et d'un vecteur.

```
def unitaire(v):
    return (1/norme(v)) * v
```

5

> Le mot-clé `assert` n'est pas au programme et sa présence dans un code d'une des premières questions du sujet a pu décontenancer un certain nombre d'étudiants. Cette instruction correspond à la vérification de la condition donnée, à savoir est-ce que `t` est positif ou non. Si c'est le cas, l'instruction ne fait rien. Si non, elle *lève* une erreur, et l'utilisateur obtient alors le message `AssertionError` avant un arrêt immédiat de l'exécution.

La fonction `pt` permet de calculer, pour un rayon lumineux `r` $= (S, \vec{u})$ et un entier `t` donnés, la position du point M du rayon tel que $\overrightarrow{SM} = t\,\vec{u}$. Elle ne fonctionne, comme indiqué dans la description donnée dans l'énoncé, que pour $t \geqslant 0$, car t est la distance séparant M de la source, dans la direction définie par $\vec{u}$.

La fonction `dir` définit le vecteur unitaire associé au vecteur $\overrightarrow{AB}$, donc au rayon passant par les points `A` et `B` et allant de `A` vers `B`.

La fonction `ra` définit le rayon lumineux partant du point `A` et passant par `B`.

6 La fonction `sp` crée la représentation de la sphère de centre `A` et passant par `B`.

```
def sp(A, B):
    return A, norme(vec(A, B))
```

7 On considère un point M appartenant à la droite passant par le point A et de vecteur directeur $\vec{u}$. Cela signifie qu'il existe un réel t tel que

$$\overrightarrow{AM} = t\,\vec{u}$$

Ce point M appartient à la sphère de centre C et de rayon r si et seulement si la distance MC vaut r, soit $\|\overrightarrow{MC}\|^2 = \overrightarrow{MC} \cdot \overrightarrow{MC} = r^2$. Or,

$$\begin{aligned}\overrightarrow{MC} \cdot \overrightarrow{MC} &= \left(\overrightarrow{CA} + \overrightarrow{AM}\right) \cdot \left(\overrightarrow{CA} + \overrightarrow{AM}\right)\\ &= \overrightarrow{CA} \cdot \overrightarrow{CA} + \overrightarrow{AM} \cdot \overrightarrow{AM} + 2\,\overrightarrow{CA} \cdot \overrightarrow{AM}\\ \overrightarrow{MC} \cdot \overrightarrow{MC} &= \|\overrightarrow{CA}\|^2 + t^2 + 2\,\overrightarrow{CA} \cdot \left(t\,\vec{u}\right)\end{aligned}$$

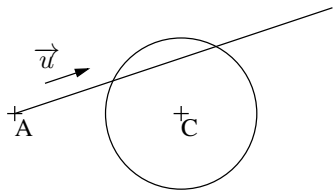
On trouve ainsi que $\boxed{t^2 + 2\,t\,\vec{u} \cdot \overrightarrow{CA} + \|\overrightarrow{CA}\|^2 - r^2 = 0}$

Cette équation est un trinôme du second degré. Si son discriminant est strictement positif, elle a deux solutions réelles correspondant aux deux intersections de la droite et de la sphère. Un discriminant nul indique une unique solution réelle double, correspondant à une droite tangente à la sphère. Enfin, dans le cas d'un discriminant strictement négatif, les solutions ne sont pas réelles, la droite et la sphère ne sont alors pas sécantes.

8 L'équation déterminée ci-dessus a des solutions réelles si

$$\Delta = 4\left(\vec{u} \cdot \overrightarrow{CA}\right)^2 - 4\left(\|\overrightarrow{CA}\|^2 - r^2\right) \geqslant 0$$

De plus, avec A à l'extérieur de la sphère, le schéma ci-contre montre que les deux solutions doivent être positives. Il faut donc que leur produit et leur somme soient positifs. Dans un trinôme de la forme $a\,x^2 + b\,x + c = 0$, le produit des racines vaut c/a et la somme $-b/a$. La condition sur le produit est déjà réalisée si A est à l'extérieur de la sphère. Celle sur la somme rend nécessaire la condition



$$\vec{u} \cdot \overrightarrow{CA} < 0$$

Si c'est le cas, il faut alors retourner la plus petite solution des deux, soit

$$t = -\vec{u} \cdot \overrightarrow{CA} - \frac{\sqrt{\Delta}}{2}$$