Examen final

Durée : 1 heure 30.

Avant de commencer

Le sujet est délibérément trop long pour vous laisser le choix des exercices abordés. Ne paniquez pas, le barême sera adapté en fonction de ce que le groupe aura réussi à faire. Ne vous concentrez pas sur les questions une par une, mais essayez de lire tout le sujet avant, car vous pourrez obtenir des indices pour certaines questions en lisant la suite. Un certain nombre de questions peuvent être répondues indépendamment des autres, donc ne restez pas coincés, essayez d'avancer rapidement sur ce que vous savez faire. Bon courage à tous.

Nous insistons sur la nécessité de **justifier toutes vos réponses** (même s'il n'est pas explicitement dit de le faire). Lorsque vous répondez simplement « oui », nous considérerons que vous avez probablement répondu au hasard, et vous ne recevrez pas les points même s'il s'agit de la bonne réponse. À l'inverse, si votre réponse est fausse, mais que votre justification fait preuve d'arguments intéressants, vous pourrez tout de même obtenir des points.

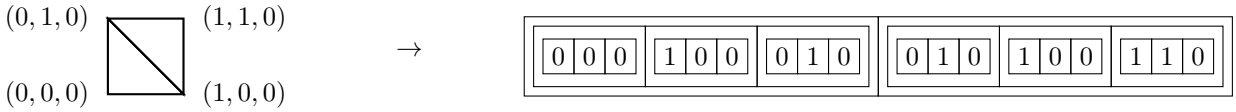
1 Structures de données maillages

Les maillages sont une représentation classique pour les objets 3D. Ces maillages peuvent être traduits de plusieurs façons en terme de structure de données. Selon les informations qui seront nécessaires aux algorithmes devant traiter les objets, les structures de données seront plus ou moins adaptées.

1.1 Tableaux simples

La façon la plus simple de stocker un maillage consiste à utiliser de simples tableaux. De façon abstraite, une telle structure peut être décrite de la façon suivante :

— un sommet est un tableau de trois réels (ses coordonnées) ;
— un triangle est un tableau de trois sommets ;
— un maillage est un tableau de triangles (dont vous connaissez la taille).



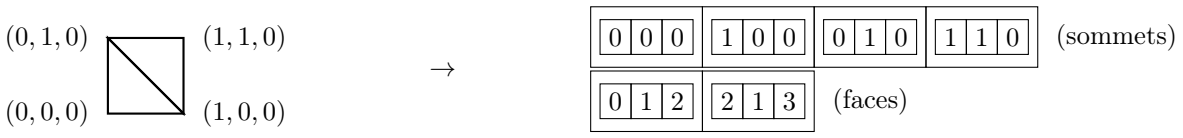
Notez bien que dans cette structure de données chaque sommet est dupliqué pour chaque triangle l'utilisant, il n'y a aucun partage.

Q1 – Avec cette structure de données, pouvez-vous appliquer une translation globale sur le maillage sans parcourir plusieurs fois le tableau ?

Q2 – Si vous ne deviez déplacer qu'un sommet du maillage en particulier (fourni par ses coordonnées), combien de triangles devrez-vous parcourir ?

Une modification classique de cette structure simple consiste à stocker les sommets indépendamment des triangles. Ainsi :

— un sommet est un tableau de trois réels (ses coordonnées) ;
— un triangle est un tableau de trois entiers (des indices de sommets dans un tableau) ;
— un maillage est un tableau de sommets et un tableau triangles (dont vous connaissez les tailles).



Q3 – Avec cette modification, donnez un exemple d'un calcul qui serait plus simple à réaliser qu'avec la structure précédente.

Q4 – Étant donné un sommet (fourni par son indice), pouvez-vous efficacement calculer la normale moyenne des faces adjacentes à ce sommet ?

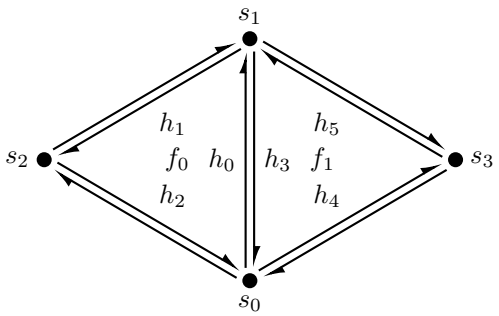
1.2 Demi-arêtes

Lorsque des traitements compliqués doivent être réalisés sur les maillages, et qu'il est nécessaire de connaître toutes les relations entre les sommets, les faces et les arêtes, une structure de données compacte classique est la structure des *demi-arêtes*. Ici, chaque face du maillage est bordée par des demi-arêtes (une par côté). Les demi-arêtes sont orientées : elles sont représentées par des flèches et non des traits. En suivant les flèches, on fait le tour d'une face dans le sens anti-horaire. Lorsque deux faces sont voisines le long d'un côté, les demi-arêtes de part et d'autre du côté (une par face) sont donc orientées en sens contraire. Par rapport aux sommets, une demi-arête est dite « entrante » si la flèche pointe sur le sommet, et « sortante » sinon. En terme de structure de données, les éléments suivants sont stockés :

Sommet :
- point : tableau de trois réels (coordonnées) ;
- arête : pointeur sur une demi-arête *sortante* (n'importe laquelle).

Face :
- arête : pointeur sur une demi-arête bordant la face (n'importe laquelle).

Demi-arête :
- sommet : pointeur sur le sommet au bout de la demi-arête ;
- face : pointeur sur la face que borde la demi-arête ;
- opposée : pointeur sur la demi-arête opposée sur la face voisine ;
- suivante : pointeur sur la demi-arête suivante sur la même face.



- s_0→arête = h_0 ou h_4 ou une autre sortante ;
- s_1→arête = h_1 ou h_3 ou une autre sortante ;
- f_0→arête = h_0 ou h_1 ou h_2 ;
- f_1→arête = h_3 ou h_4 ou h_5 ;
- h_0→sommet = s_1 ;
- h_0→face = f_0 ;
- h_0→suivante = h_1 ;
- h_0→opposée = h_3 ;

Quelques exemples d'algorithmes :

```
Fonction sommetsArete (h) → paire de sommets
    // renvoie les deux sommets d'une arête
    retourner (h→sommet,
     h→opposée→sommet)
```

```
Fonction sommetsFace (f) → ensemble de sommets
    // renvoie tous les sommets d'une face
    S ← ∅, début ← f→arête, curseur ← début
    répéter
        ajouter curseur→sommet à S
        curseur ← curseur→suivante
    jusqu'à curseur == début
    retourner S
```

Q5 – À la manière des exemples, pouvez-vous écrire une fonction qui liste les faces voisines d'une face ?

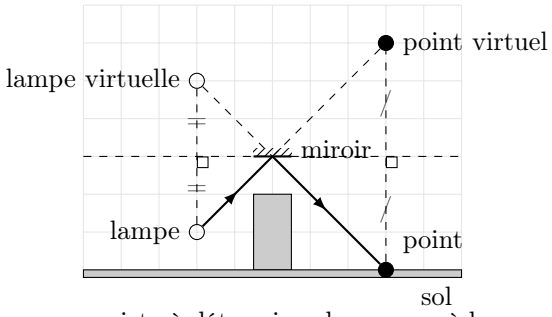
Q6 – Pouvez-vous écrire une fonction qui liste les sommets voisins (reliés par une arête) d'un sommet ?

Q7 – Pouvez-vous donner un exemple d'un calcul plus simple à réaliser avec cette structure de données qu'avec les précédentes à base de tableaux ?

Q8 – (Dur) Comment étendriez-vous la structure pour gérer les maillages à bords (certains côtés de certaines faces n'ont pas de face voisine) ? Et l'algorithme listant les sommets voisins d'un sommet ?

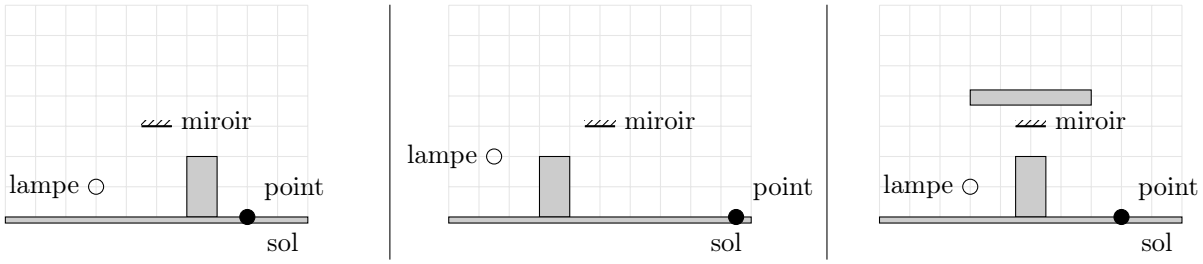
2 Miroirs

Cet exercice a pour but d'explorer quelques techniques couramment utilisées pour le rendu en présence de miroirs. Dans ce contexte, pour les miroirs plans, une technique couramment utilisée est celle des objets virtuels. Un objet virtuel est le symétrique par rapport au plan du miroir d'un objet réel. Ci dessous, vous avez par exemple un exemple d'utilisation de lampe virtuelle et de point virtuel pour calculer un chemin lumineux possible entre un point et une lampe ponctuelle. Les objets gris sont des obstacles opaques.



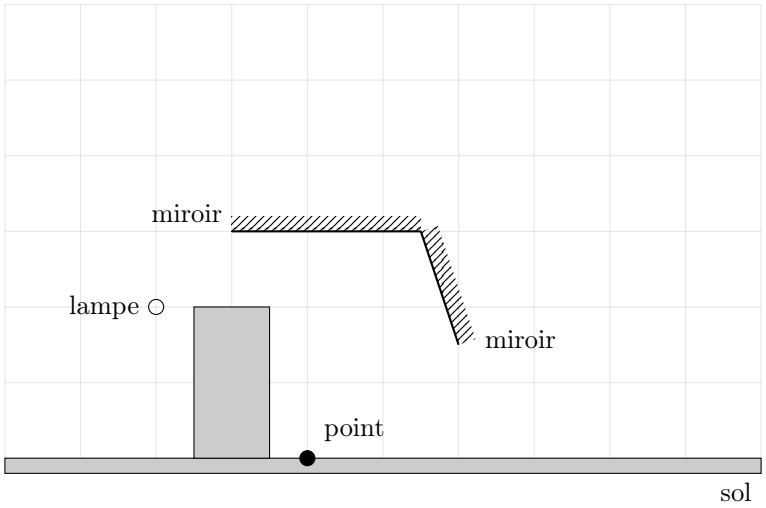
Votre but dans un premier temps consiste à déterminer les rayons à lancer et les intersections à tester pour déterminer si un objet est éclairé ou non par une lampe ponctuelle.

Q9 – Dans les trois cas illustrés ci-dessous, indiquez si le point est éclairé ou non par la lampe via le miroir. Vous pouvez dessiner sur le sujet, mais n'oubliez pas de l'intégrer à la copie rendue en précisant le numéro de copie. Vos dessins n'ont pas besoin d'être très précis, normalement il ne devrait pas y avoir d'ambiguïté.



Q10 – En déduire les rayons à lancer et les tests à réaliser pour déterminer si un point est éclairé par une lampe ponctuelle via un miroir plan. Vous préciserez les origines et directions des rayons, ainsi que les intersections cherchées pour déterminer la visibilité.

Q11 – Dans le schéma ci-dessous, pouvez-vous étendre votre approche au cas de deux miroirs et tracer un chemin lumineux entre la lampe et le point ?



Q12 – Si l'on suppose que le chemin lumineux est limité à deux rebonds sur des miroirs, pouvez vous indiquer, dans le cas de deux miroirs, les rayons que vous lanceriez pour tester si un point est éclairé par une lampe via un ou deux rebonds sur les miroirs ?

Q13 – Dans le cas de deux miroirs, le nombre de rebonds est-il toujours au maximum deux ? Si ce n'est pas le cas, pouvez-vous fournir un maximum de rebonds ? Illustrez votre réponse par un ou des exemples.

3 Phares

Les questions suivantes permettent de compléter le travail réalisé en tp : une course avec au moins 2 voitures sur un terrain. L'objectif est d'écrire le fragment shader capable d'éclairer le terrain soit par les phares des voitures, soit par le soleil.

C'est la nuit, on veut éclairer la route avec un phare d'une voiture. On considère que l'espace éclairé par le phare devant la voiture peut être représenté par un cône. Le cône sera décrit par un sommet, un axe (une direction) et un angle d'ouverture (ou son cosinus).

Q14 – Comment déterminer qu'un point se trouve dans la zone éclairée par le phare ? Détaillez votre solution.

Q15 – Comment déterminer qu'un point n'est pas éclairé ? Détaillez votre solution.

Q16 – Dans quel repère faire les calculs géométriques ? Est ce que plusieurs choix sont possibles ?

Q17 – Il y a, en général, 2 phares sur une voiture. Comment adapter votre solution à ce cas ?

Q18 – Il y a, en général, plusieurs voitures sur le terrain. Comment adapter votre solution à ce cas ?

C'est le jour, le soleil éclaire le terrain et les voitures.

Q19 – Comment représenter le soleil ? Comment déterminer qu'un point du terrain est à l'ombre d'une voiture ? Quelles informations sont nécessaires ? Est ce que cette solution représente beaucoup de calculs ?

Q20 – Pour simplifier le shader, on souhaite faire le même type de calcul que dans le cas du phare. Comment représenter le soleil dans ce cas ? Comment adapter votre solution précédente ?

Q21 – Toujours pour simplifier le shader, on souhaite limiter au maximum la quantité de calculs. Si l'on décide de remplacer la géométrie exacte de la voiture par une forme plus simple, comment adapter votre solution ? Quel sera l'impact visuel de ce changement ? Quelle forme simple semble bien adaptée ? Savez vous calculer (l'existence) de l'intersection de cette forme avec une demi-droite (ou un rayon) ? Détaillez votre solution.

Q22 – C'est la nuit, comment déterminer qu'un point du terrain est à l'ombre d'une voiture éclairée par un phare ?

Q23 – Bonus : comment calculer la couleur du point du terrain ou d'une voiture, sachant qu'il est éclairé ? Même question, si le point est à l'ombre ?

Q24 – Quelles informations doit connaître le fragment shader pour fonctionner et utiliser votre solution pour déterminer qu'un fragment est à l'ombre ou éclairé ? Comment lui transmettre ces informations ?

Q25 – Bonus : écrivez le fragment shader.