TD 12 : Transformations

Thomas Aynaud et Areski Cousin

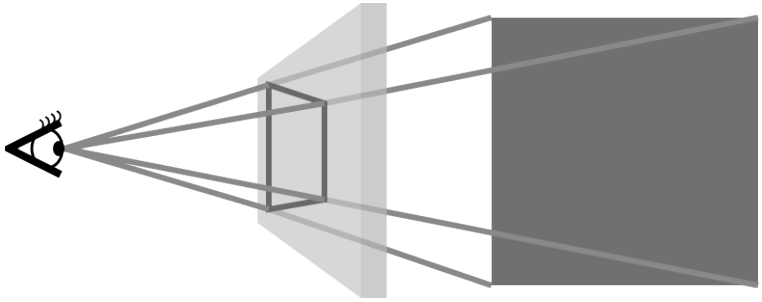
29 avril et 5 mai

1 Matrices de rotation

1. Si f est la rotation d'axe orienté par v_0 et d'angle θ, quelle est l'image d'un vecteur v par f ?
2. Quelles sont alors les coordonnées des images des vecteurs de la base canoniques de $\Re^3$?
3. Ecrire une fonction `gen_rotation` prenant en entrée un vecteur v_0 et un angle θ et renvoyant la matrice de rotation d'axe orienté par v_0 et d'angle θ
4. Calculer la matrice de rotation d'axe $(1, -1, 1)$ et d'angle $\frac{\Pi}{3}$
5. Calculer son déterminant, sa transposée et son inverse.
6. On suppose maintenant avoir une matrice de rotation M. Ecrire une fonction `angle` et une fonction `direction` renvoyant l'angle et la direction d'une matrice de rotation.

2 Projection perspective

Il existe plusieurs moyen de représenter un objet 3D sur un écran 2D. L'un d'eux consiste à passer par une projection que l'on appelera la projection perspective. Le principe est de trouver le point d'intersection du rayon lumineux partant de l'objet jusqu'à l'oeil de l'observateur avec un plan particulier, qui sera l'écran :



1. Faire un dessin en vue "de coté" en considérant que l'écran est le plan Oxy et que l'observateur est le long de l'ax Oz à distance d du point O. Appliquer le théorème de Thalès pour calculer les coordonnées du projeté d'un point en fonction de d.
2. Peut-on directement faire cette transformation par une multiplication de matrice ?
3. Ecrire une procédure `projeteP(p,d)` projetant un point donné sous la forme d'une séquence de trois réels, en supposant qu'il est du bon coté du plan 0xy...
4. Ecrire une procédure `projete(s,d)` projetant une séquence de points.
5. Ecrire une procédure `translateP(p,v)` translatant un point suivant le vecteur v.
6. Ecrire une procédure `translate(s,v)` translatant une séquence de points.
7. Ecrire une séquence de points permettant de décrire les arrêtes d'un cube.
8. Utilisez les fonctions précédentes (exercice précédent compris) pour afficher avec plot un joli cube projeté.

3 Triangle de Sierpiński

Voici les premiers termes d'une suite de dessins appelés les triangles de Sierpiński :

C'est un dessin fractal, appelé triangle de Sierpiński. Nous allons implémenter une procédure traçant le triangle à l'itération n.

1. Écrire une séquence `s` telle que `plot(s)` affiche un triange équilatéral.
2. Écrire sur une feuille comment passer d'un triangle au suivant avec des copies, des translations et des homothéties.
3. Écrire une procédure `translation(s,v)` prenant en paramètre une séquence s de points et un vecteur v sous la forme de deux coordonnées et renvoyant la séquence des translatés des points de s suivant v.
4. Écrire une procédure `reduire(s)` prenant en paramètre une séquence s de points et leur appliquant à chacun l'homotétie de centre O et de rapport $\frac{1}{2}$.
5. Écrire une procédure `suivant(s)` prenant une séquence s décrivant le n-ième triangle de sierpinski et renvoyant la séquence décrivant le triangle suivant.
6. Écrire une procédure `sierpinski(n)` utilisant les fonctions précédentes et renvoyant les points du n-ieme triangle de Sierpiński et l'afficher.

4 Orthonormalisation de Gram-Schmidt

En algèbre linéaire, le procédé de Gram-Schmidt est une méthode pour orthonormaliser une famille libre de vecteurs d'un espace vectoriel muni d'un produit scalaire. À partir d'une famille libre $(v_1, ..., v_n)$, on construit une famille orthonormale $(e_1, ..., e_n)$, qui engendre les mêmes espaces vectoriels successifs :

$$\forall j < n, F_j = \text{Vect}(e_1, \ldots, e_j) = \text{Vect}(v_1, \ldots, v_j).$$

Le procédé de Gram-Schmidt est :

$$\mathbf{u}_1 = \mathbf{v}_1$$

$$\mathbf{e}_1 = \frac{\mathbf{u}_1}{||\mathbf{u}_1||}$$

$$\mathbf{u}_k = \mathbf{v}_k - \sum_{j=1}^{k-1} \text{proj}_{\mathbf{u}_j}(\mathbf{v}_k)$$

$$\mathbf{e}_k = \frac{\mathbf{u}_k}{||\mathbf{u}_k||}$$

Implémentez le procédé d'orthonormalisation de Gram-Schmidt.