Projet de Programmation C

guillaume.revy@univ-perp.fr

Visualisation 3D d'un objet convexe

Partie 1 Objectif du projet

L'objectif de ce projet est de réaliser un programme permettant de visualiser en 3D un objet convexe composé de faces. Dans un premier temps, on s'intéressera à visualiser un cube. Ce programme devra être réalisé en utilisant la bibliothèque graphique SDL2.

Ce projet est à réaliser en **binôme maths/info**, et à déposer par un des deux étudiants sur le Moodle au plus tard le **5 décembre 2017 à minuit**.

Partie 2 Modélisation d'un objet convexe

Un objet convexe est composé d'un nombre f de faces. Chaque face sera, quant à elle, un polygône ayant un nombre s de sommets. Dans notre cas, un cube sera composé de $f = 6$ faces à $s = 4$ sommets, et il aura des arêtes de longueur a.

En infographie, pour faciliter les opérations de rotation et de translation (voir Partie 5), et pouvoir les effectuer via des opérations matricielles, chaque sommet est représenté en coordonnées homogènes $(x, y, z, 1)$.

► **1.** Définir une structure polygone_t qui permet de modéliser une face à s sommets.

Dans cette structure, les sommets d'une face devront être ordonnés dans le sens direct : ceci sera utile lors de la gestion des faces cachées (voir Partie 4).

► **2.** Définir une structure objet_t qui permet de modéliser un objet convexe à f faces.

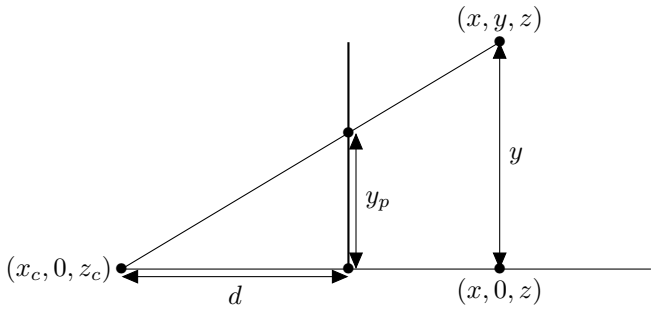
► **3.** Écrire une fonction creer_cube, qui prend en paramètre les coordonnées (x, y, z) d'un sommet du cube et la longueur a des arêtes, puis qui créé et renvoie l'objet après avoir déterminer, pour chaque face, les coordonnées de chaque sommet.

Partie 3 Affichage d'un objet 3D

La première étape consiste à afficher l'objet 3D par projection sur un plan 2D. Pour ce faire, on considère que la caméra se situe en coordonnées (x_c, y_c, z_c), que le plan de projection se situe à une distance d de la caméra, et que le coordonnées de tous les sommets de l'objet vérifient $z > z_c + d$.

► **1.** Définir une structure camera_t qui permet de modéliser la position de la caméra, c'est-à-dire, ses coordonnées (x_c, y_c, z_c) et la distance d au plan de projection.

Ensuite, les coordonnées (x_p, y_p) du projeté d'un sommet (x, y, z) peuvent être déterminées en utilisant le théorème de Thalès, comme l'illustre la figure ci-dessous.



► **2.** Écrire une fonction projeter_sommet qui, étant donnés la position de la caméra et les coordonnées (x, y, z) d'un sommet, calcule et renvoie les coordonnées du projeté (x_p, y_p).

► **3.** Écrire une fonction afficher_cube qui prend en paramètre le cube et une fenêtre graphique, et qui permet d'afficher le projeté 2D de l'objet 3D.

Partie 4 Gestion des faces cachées

Maintenant que l'objet est affiché, l'étape suivante est de modifier le programme pour ne plus afficher les faces cachées. Pour déterminer si une face est cachée, le processus est le suivant :

1. en utilisant les coordonnées de trois points de la face, déterminer deux vecteurs du plan formé par la face, notés $\overrightarrow{v_1}$ et $\overrightarrow{v_2}$)
2. déterminer un vecteur normal du plan, noté $\overrightarrow{v}$, orienté vecteur l'intérieur du volume, en calculant le produit vectoriel de $\overrightarrow{v_1}$ et $\overrightarrow{v_2}$,
3. et calculer le produit scalaire entre le vecteur normal $\overrightarrow{v}$ et le vecteur $\overrightarrow{OS}$ où O est la position de la caméra et S un sommet de la face : la face n'est visible que si ce produit scalaire est strictement positif.

► **1.** Écrire une fonction qui, étant données une face du cube et la position de la caméra, détermine si cette face est visible.

► **2.** Modifier la fonction afficher_cube précédente, pour n'afficher que les faces visibles.

Partie 5 Translation et rotation

La dernière étape consiste à appliquer des translations et rotations à notre objet. Comme indiqué précédemment, ces deux transformations peuvent être effectuées via des opérations matricielles. Plus particulièrement, appliquer une tranformation à un objet revient à appliquer cette même transformation à chaque face de l'objet, sommet par sommet.

Translation. Soit un sommet de coordonnées $(x, y, z, 1)$. Appliquer une translation de vecteur (t_x, t_y, t_z) à ce sommet consiste à effectuer le produit matriciel suivant :

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

où $(x', y', z', 1)$ sont les nouvelles coordonnées du sommet.

► **1.** Écrire une fonction multiplication_matrice qui prend en paramètre une matrice 4×4 et un vecteur de taille 4, et qui calcule et retourne le résultat du produit matrice-vecteur correspondant.

► **2.** Écrire une fonction translation_cube, qui étant donnés un cube et les coordonnées du vecteur de translation, calcule les nouvelles coordonnées de chaque sommet du cube.

Rotation. Soit un sommet de coordonnées $(x, y, z, 1)$. Appliquer une rotation d'angle θ à ce sommet consiste à effectuer le produit matrice-vecteur suivant :

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = R \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}, \quad \text{avec} \quad R \in \{R_x, R_y, R_z\}$$

où $(x', y', z', 1)$ sont les nouvelles coordonnées du sommet, et R_x, R_y, et R_z sont les matrices de rotation par rapport l'axe (0_x), (0_y), et (0_z), respectivement, définies de la manière suivante :

$$R_x = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & \sin(\theta) & 0 \\ 0 & -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad R_y = \begin{pmatrix} \cos(\theta) & 0 & \sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) & \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad R_z = \begin{pmatrix} \cos(\theta) & \sin(\theta) & 0 & 0 \\ -\sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

► **3.** Écrire une fonction rotation_cube, qui étant donnés un cube, l'angle de rotation et l'axe de rotation, calcule les nouvelles coordonnées de chaque sommet du cube.