Simulation d'un projecteur

Thibaud Lambert, Brett Ridel

L'objectif de ce TP est de simuler un projecteur virtuel dans une scène 3D (Figure 2). Le projecteur envoie sur les objets de la scène une image (une texture 2D). Une image résultat est visible sur la figure 1.

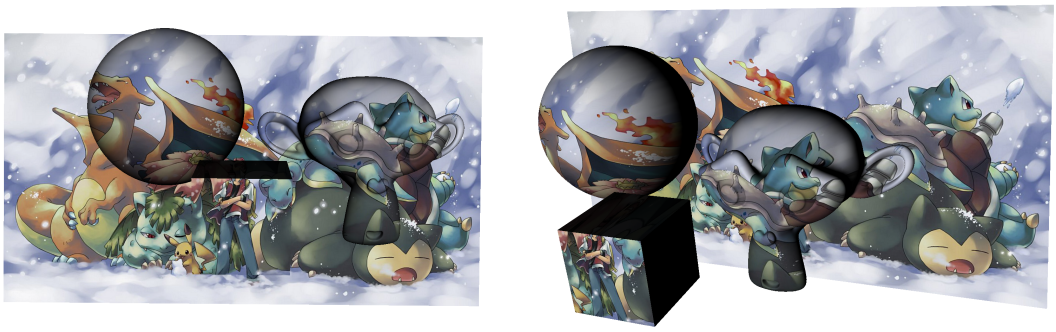

Figure 1 – Résultat attendu lorsque le projecteur et la caméra sont presque alignés (gauche) et lorsqu'ils sont désaxés (droite).

Le projet fournit permet de charger et afficher un maillage. Une trackball est fournie vous permettant de modifier l'orientation de la caméra. Du code est également fournit pour charger une texture 2D (voir **Viewer::loadTexture()**) et l'envoyer à OpenGL (voir **Viewer::sendTexture()**). Le code actuel affiche un objet 3D avec ses normales.

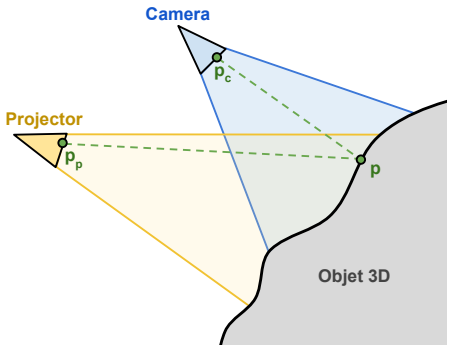

Figure 2 – Simulation d'un projecteur. Un point p de l'objet 3D, éclairé par un projecteur, est visualisé à travers une caméra. Le point p correspond à la projection du point p_p confondu dans le plan image du projecteur, et est projeté en p_c dans le plan image de la caméra.

En considérant le projecteur et l'objet fixes, l'image projetée par le projecteur doit être appliquée sur l'objet. D'un point de vue implémentation, on peut considérer le projecteur comme une caméra, c'est à dire un système qui possède une position, une orientation et une matrice de

projection. Dans le classe **Viewer**, créez un objet **Camera _projector**.
Pour son initialisation (dans **Viewer::init**), utilisez les méthodes **Camera::setPerspective** et **Camera::lookAt**. Calculez l'aspect ratio en fonction des dimensions de l'image accessibles avec **_image.width()** et **_image.height()**. Pour le lookat, vous pouvez utiliser une position de $(0.5, 1.5, 7.5)$ et un target de $(0, 0, 0)$.

Pour accéder une texture dans un shader, utilisez la ligne de code suivante **uniform sampler2D tex_name**. Pour le nom de la texture, utilisez celui employé dans le code d'envoie de **Viewer::sendTexture**. Pour récupérer la valeur d'un pixel dans une texture en GLSL, utilisez la fonction suivante **texture(sampler2D tex_name, vec2 uv_coord)** avec **uv_coord** compris entre $(0, 0)$ et $(1, 1)$.