UJF - M2 IICAO 2005-2006 : Infographie
TP6 : OpenGL avancé

15 novembre 2005

Aujourd'hui nous allons effectuer un filtrage "*antialising*", une méthode permettant d'obtenir un effet de profondeur de champ et l'utilisation d'une méthode de pochoir.

1 Antialiasing

1.1 Principe

Vous avez dû vous rendre compte que les coins et arêtes vives des objets, en openGL, apparaissent crénelés, ceci s'appelle l'*aliasing*. Dans notre cas, ceci provient du fait que la "projection" des objets sur le plan image en 3D est en fait répercuté sur une grille : l'image en pixels. Ces pixels sont des valeurs entières. Si l'arête d'un objet "tombe" à l'intérieur d'un certain pixel, OpenGL n'en sait absolument rien et ne dessine qu'un côté ou l'autre de cette arête.

1. Vérifiez cet artefact en compilant et éxecutant le programme. Les touches X (et x) et Y (et y) permettent de décaler légèrement les plans left-right et bottom-top du frustum (ne vous souciez pas du code pour l'instant). Vous devriez observer des "sauts" de couleur sur les arrêtes du cube jaune.

Cet effet est assez génant visuellement. Il existe plusieurs manières de réduir ce défaut, une des manières consiste à prendre plusieurs images en décalant légèrement l'image (donc les pixels) dans les 2 directions, sans changer la position de la caméra.

La procédure definirCamera prend maintenant trois paramètres, intéressons nous aux deux premiers : le nombre de pixels (attention c'est un réel, on peut donc ici parler de un demi pixel) en x et le nombre de pixels en y pour lesquels il faut décaler l'image. Si ils valent (0.5,0), cela signifie que l'on décale le plan image (via le frustum) de 0.5 pixels vers la droite et de 0 pixel vers le haut, dans l'espace.

1.2 Mise en oeuvre

Si l'on dessine plusieurs fois la scène en décalant à chaque fois la caméra via ces paramètres et en mélangeant les images obtenues, cela aura un effet de

floutage, d'antialising. L'utilisation d'un autre buffer que le buffer d'image, le buffer d'accumulation, permet de mélanger plusieurs images en les ajoutant.

En voici le principe :

1. Effacer le buffer d'accumulation : `glClear(GL_ACCUM_BUFFER_BIT);`
2. Pour chaque image désirée de la scène que l'on va rendre :
   (a) Effacer le buffer d'image et le buffer de profondeur pour pouvoir dessiner une nouvelle image : `glClear( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);`
   (b) Dessiner la scène comme d'habitude en ayant défini au préalable la caméra de la manière souhaitée (pour l'antialiasing).
   (c) Ajouter l'image dans l'accumulateur en divisant la valeur des couleurs par M (normalement l'inverse du nombre d'images) : `glAccum(GL_ACCUM, M);`
3. Recopier le buffer d'accumulation vers celui de l'image : `glAccum (GL_RETURN, 1.0);`

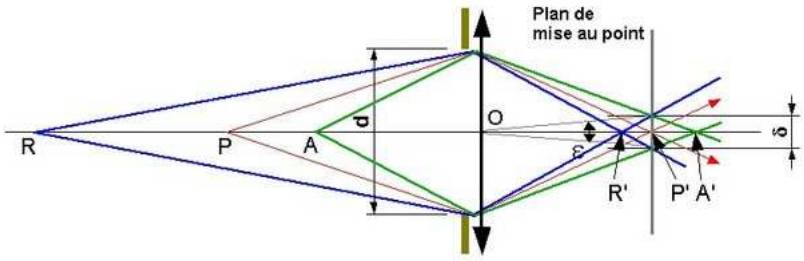
Utilisez ce principe pour effectuer un lissage de l'image (cf. type "`ANTIA`", utilisez la touche 't' pour changer de type). Pour cela, complétez la procédure display pour le type "`ANTIA`" (et "`DOF`" par la même).
Attention : l'utilisation (même très simple) de ce buffer fait ramer l'application (ainsi, réduisez la taille de la fenêtre si nécessaire pour que ça aille un peu plus vite).

Bien sûr, il suffirait d'appliquer un filtrage gaussien à l'image résultat tout seule pour obtenir ceci, mais l'utilité de ce buffer va plus loin. Ainsi il permet facilement de créer une application de visualisation stéréoscopique (avec les lunettes de couleur ridicules), du motion blur, ou par exemple, de créer un effet de profondeur de champ.

2 Profondeur de champ

2.1 Principe

Un appareil photo possède un système de mise au point automatique ou pas : le focus (d'où " l'autofocus") qui permet de régler le plan de mise au point. Les objets plus loin ou plus proche d'une certaine distance (la distance de mise au point) apparaissent flous.



2.2 Visualisation et explication

Il est possible de simuler naïvement un tel rendu simplement en déplaçant le frustum ET la position dans l'espace de la caméra.

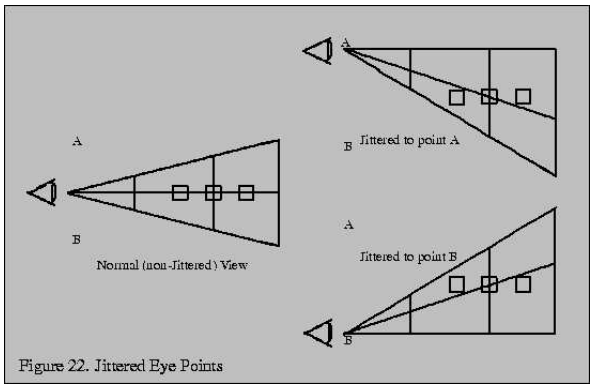


Figure 22. Jittered Eye Points

Le type d'éxécution "`DOF`" (pour Depth Of Field, profondeur de champ) permet d'obtenir ce résultat en se basant sur la méthode d'antialiasing précédente (vous n'avez rien à changer dans le code à part mettre un facteur "0.01" devant les deux premiers paramêtres donnés à `definirCamera`).

Pour changer la distance de mise au point (le focus), utilisez les touches 'f' et 'F'.

1. Rapprochez le jusqu'à 0.5, qu'observez vous ? Eloignez le vers les 15, même question.
2. D'après le code déjà existant (où l'on différentie simplement "`ANTIA`" de "`DOF`" dans la procédure `definirCamera`) et les schémas précédents, comment expliquez-vous le fonctionnement de cette méthode ?

3 Technique de pochoir

En dessin, un pochoir est (par exemple) une feuille découpée d'un motif; pour reproduire ce motif sur une toile, il suffit alors de poser le pouchoir aux endroits désirés et de peindre par dessus.

En OpenGL, le *stencil buffer* permet d'effectuer des opérations similaires. Executez le programme avec le type "`STEN`" : vous devriez voir dessinés la même scène qu'avant, mais comme vue à travers un objectif d'appareil photo, comprennant ici 2 cercles où le petit comprend la scène zoomée en plus petit et une croix noire au milieu.

Pour effectuer cette manip', il a suffit de dessiner deux cercles et une croix dans le stencil buffer et ensuite d'indiquer (cf. `display`) à openGL quoi dessiner dans quel cercle (la création du pochoir est effectuée dans `definirPochoir`, jetez-y un oeil si vous voulez voir plus en détail comme il est fait).

1. Servez vous de l'accumulateur pour dessiner la scène deux fois plus claire à travers le petit cercle que dans le cercle principal.

Le stencil buffer est souvent utilisé pour effectuer des ombres portées ou pour la réflection d'une scène dans un mirroir.