

Pour des applications plus complexes, OpenGL permet de séparer le traitement des couleurs (r,g,b) de celui du canal alpha. C'est-à-dire qu'il y a une équation de mélange pour les trois composantes de couleur, avec ses facteurs multiplicatifs et une autre équation avec des facteurs différents pour le canal alpha. On met en place un tel mélange distinct avec les fonctions `glBlendEquationSeparate(operateurRGB, operateurAlpha)` et `glBlendFuncSeparate(sfactorRGB, dfactorRGB, sfactorAlpha, dfactorAlpha)`. Nous en verrons un exemple dans la [Section 20.3, Transparence en arrière](#).

Voyons maintenant les applications de ces possibilités de mélange, en commençant par les plus faciles à comprendre.

20.2. Transparence en avant

Nous allons nous intéresser à la superposition d'éléments transparents devant des éléments dessinés précédemment. C'est ce qui paraît le plus simple. Nous verrons ultérieurement qu'il est possible de dessiner un élément opaque derrière un élément transparent dessiné auparavant.

Dans toute la suite, je vais travailler avec des images semi-transparentes telles que celle de la figure suivante. Ces images sont transparentes à cause de leur canal alpha. Seuls les fichiers PNG contiennent un canal alpha. Les fichiers JPG n'en contiennent pas et ne sont pas utilisables ici.

Figure 20.3 : Image semi-transparente



J'ai dessiné son canal alpha à droite. Là où il est noir, l'opacité vaut 0, c'est-à-dire que l'image est transparente. Inversement, le blanc indique une opacité totale. Cette image ne montre pas de zones intermédiaires, mais c'est évidemment possible.

L'image suivante a été dessinée par le projet [Blending](#) que je vous invite à consulter. Ce projet est bâti autour de deux classes supplémentaires :

- la classe `Rectangle` définit un maillage de quatre sommets portant des coordonnées de texture. Ce rectangle est associé à une texture ;
- la classe `Scene` crée des rectangles et les dessine.

Figure 20.4 : Empilement d'éléments transparents



Cette scène est composée de 4 rectangles sur lesquels je plaque 4 images. Au fond, il y a la colline. Au dessus, il y a des nuages, ensuite un palmier et au premier plan un arbuste. Pour les rendre transparents, j'utilise le mode de mélange de la section précédente : $résultat = \alpha_{source} * source + (1 - \alpha_{source}) * destination$. Là où le canal alpha de la source vaut 0, on voit uniquement la destination, c'est-à-dire ce qui était déjà dessiné. Là où le canal alpha vaut 1, on ne voit que la source, c'est-à-dire l'image en cours de dessin.

Si on n'active pas le mélange, le canal alpha n'est pas pris en compte, et chaque rectangle reste opaque, en écrasant ce qui est derrière. Pour le constater, il vous suffit de mettre en commentaire l'instruction `gl.enable(gl.BLEND)`.

Le seul point à comprendre, c'est qu'il faut dessiner l'arrière-plan en premier, puis les nuages, puis le palmier, puis l'arbuste. Chacun se mélange avec celui qui est derrière. Si les objets ne se superposent pas, on peut les dessiner dans l'ordre qu'on veut, mais si l'un est devant l'autre, alors il faut le dessiner après. Amusez-vous à détériorer le programme en changeant l'ordre d'affichage des éléments, dans la méthode `onDraw` de la classe `Scene`. Vous pouvez aussi changer la coordonnée z de la translation qui les positionne.

Dessin de panneaux

L'exemple précédent m'amène à présenter le concept de panneau (*billboard*). C'est une simplification extrême d'un objet complexe lointain. Le mot panneau fait référence aux panneaux publicitaires au bord des routes. La [Figure 20.5](#) a été dessinée à l'aide d'un panneau, basé sur une image d'arbre ([bb_arbre2_8.png](#)). Une telle image peut être insérée dans un paysage en remplacement d'un modèle 3D très complexe. Au lieu de dessiner des milliers de polygones, on ne dessine qu'un simple rectangle.

Figure 20.5 : Panneau



L'image de l'arbre peut avoir été créée dans un FBO par une synthèse d'images préalable à partir d'un modèle complexe. L'intérêt est qu'ensuite, elle ne change pas d'une image à l'autre, parce qu'on est censé la voir de relativement loin.

Par rapport aux rectangles du précédent exemple, il y a deux points spécifiques concernant les panneaux :

- les panneaux doivent toujours rester face à la caméra, même si on tourne autour. On ne doit pas voir un panneau de profil, parce qu'on verrait immédiatement la supercherie ;
- on peut gérer un ensemble de textures pour offrir une image différente quand on tourne autour d'un panneau.

Voici les solutions technique à ces points.