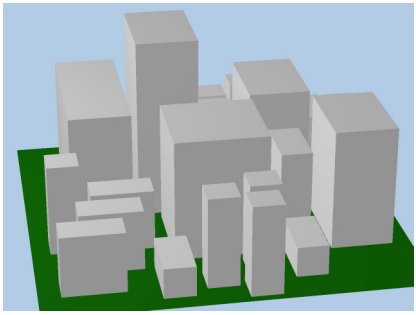
19.4. Occlusion ambiante

Pour finir ce chapitre, je vais vous expliquer une technique utilisée dans les jeux vidéo depuis quelques années. Elle concerne l'éclairement global d'une scène, également appelé éclairement ambiant. Voici une scène éclairée par deux lampes. Une lampe omnidirectionnelle située vers la gauche et en arrière. Elle provoque la différence d'éclairement des faces des cubes. La seconde lampe les éclaire d'une manière uniforme. C'est ce qui fait que le dos des cubes n'est pas noir.

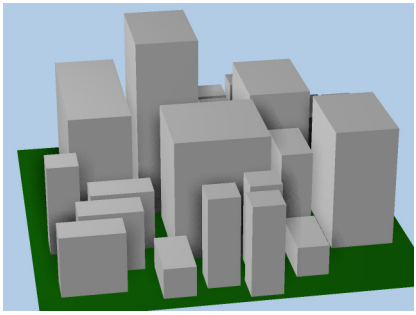
Figure 19.9 : *Éclairement ambiant*



Pour constater l'effet de la seconde lampe, il vous suffit de commenter l'ajout de son éclairement, `addLighting(m_Light2)`, dans la méthode `onDrawFrame` de la classe `Scene` du projet `Ambiant`. Inversement, vous pouvez commenter l'autre appel à `addLighting` et ajouter `true` en paramètre à celui de la deuxième lampe. Notez que la lampe omnidirectionnelle est animée.

La scène est donc éclairée, mais le résultat manque de relief. Je vous propose d'étudier une amélioration qui conduit au résultat suivant, beaucoup plus réaliste. Les programmes sont dans le dossier `SSAOShadows`.

Figure 19.10 : *Éclairement SSAO*



Ce résultat présente des défauts, nous verrons pourquoi, mais il ajoute des sortes d'ombres dans les espaces confinés entre les cubes. Ces endroits sont peu éclairés parce qu'il y a des objets tout autour. C'est ce qu'on appelle l'occlusion ou obstruction globale. Elle provient du fait que le ciel qui apporte la lumière est relativement peu visible, il est masqué. Si on suivait les lois de l'optique et de la géométrie, il faudrait calculer la proportion de ciel visible en tout point du maillage, mais c'est trop compliqué pour le faire en temps réel. D'autant que l'arrière des objets ne nous intéresse pas.

La méthode SSAO est une simplification du calcul de la visibilité du ciel. Le sigle signifie *Screen Space Ambiant Occlusion*, c'est-à-dire obstruction ambiante calculée dans le plan de l'écran. Il existe plusieurs versions plus ou moins simples. Le principe général ressemble à celui de la section précédente, les ombres PCSS. Le shader compte les occulteurs dans le voisinage du fragment. La grosse différence est que la recherche ne se fait pas dans la carte d'ombre de la lampe, puisqu'il n'y en a pas, mais dans le Depth Buffer de l'image elle-même, ou bien, en dessin différé, dans le buffer des positions qui est plus précis. Ce buffer contient les distances entre les objets les plus proches et la caméra. Lorsqu'on étudie le voisinage d'un point dans ce buffer, on peut rencontrer des distances plus faibles que celle du point courant. Dans ce cas, le point est derrière d'autres objets. On peut aussi trouver des distances plus grandes qui indiquent que le point est devant. Le principe est de ne s'intéresser qu'aux objets qui sont devant, qui peuvent masquer la lumière, en tenant compte de la normale de la surface au niveau du point considéré.

Voici le code source GLSL de cette lampe. Il y a plusieurs astuces qui sont signalées par les numéros.