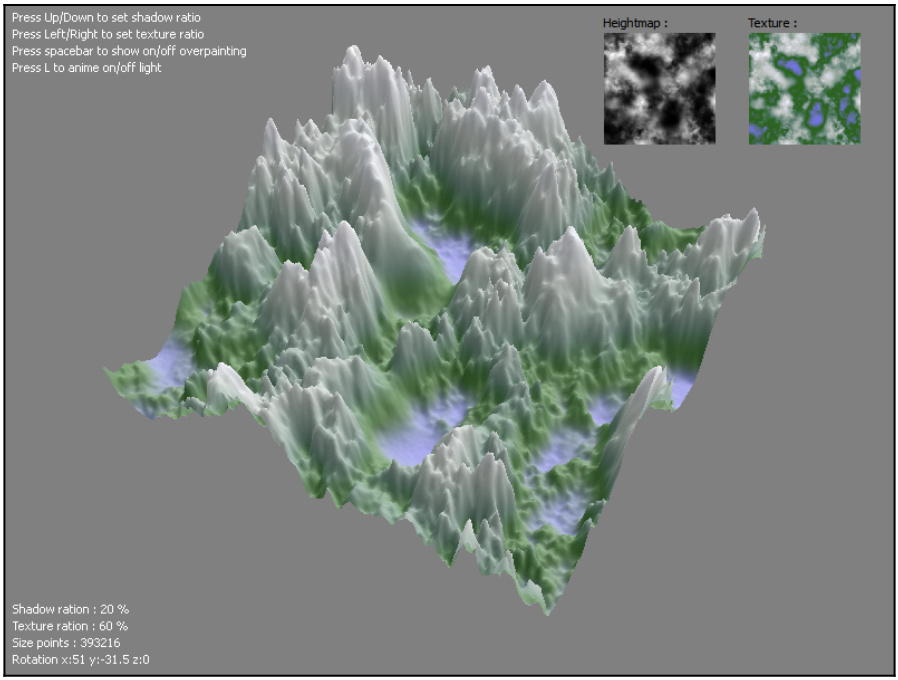
Qt OpenGL - Overpainting : dessiner en 2D avec QPainter sur une scène 3D

Sommaire

Dans certaines applications, il peut être utile de dessiner des objets en premier plan d'une fenêtre de rendu OpenGL (par exemple un affichage tête haute pour les jeux vidéo). Dans cette partie, nous allons afficher en premier plan de la vue 3D les images de la heightmap et de la texture utilisées pour la génération du terrain.



Qt dispose d'une classe dédiée au dessin 2D très complète : QPainter. Cette classe permet de dessiner toutes sortes de formes, de la simple ligne aux polygones complexes. Elle est aussi capable de dessiner des images à l'aide des fonctions drawImage et drawPixmap. Dans cet exemple, nous allons utiliser la fonction drawPixmap pour afficher les images et la fonction drawText pour afficher des informations. Pour rappel, la classe QPixmap est optimisée pour l'affichage, contrairement à QImage qui est optimisée pour la manipulation de pixels.

Habituellement, pour réaliser un rendu 3D, on n'a pas besoin d'implémenter la fonction paintEvent. Celle-ci est appelée automatiquement lors de la mise à jour du rendu et appelle la fonction paintGL. Pour réaliser l'overpainting, nous allons devoir modifier cette fonction paintEvent et appeler manuellement la fonction paintGL. Il faut donc dans un premier temps modifier le code de notre timer pour que celui-ci appelle la fonction update à la place de la fonction updateGL :

```
    connect(&timer, SIGNAL(timeout()), this, SLOT(update()));
```

```
}
```

Ensuite, il faut spécifier à QWidget que l'on prend en charge la mise à jour du background et qu'il n'est pas nécessaire de le faire automatiquement. Pour cela, on ajoute les deux lignes suivantes dans le constructeur :

```
    setAttribute(Qt::WA_OpaquePaintEvent);
    setAutoFillBackground(false);
}
```

Dans la fonction paintEvent, on commence par appeler la fonction de rendu 3D OpenGL paintGL. Ensuite, on crée un QPainter pour dessiner sur le widget courant. Lorsque le widget courant est un QGLWidget, QPainter utilise automatiquement OpenGL pour faire le rendu 2D. Lors de la création du contexte de QPainter, les paramètres 3D sont donc modifiés : il est donc nécessaire de les réinitialiser dans paintGL (par exemple d'activer le test de profondeur) :

```
    glEnable(GL_DEPTH_TEST);
```

Normalement, il faudrait aussi redonner les matrices de transformation 3D. Dans notre cas, les matrices sont envoyées directement aux shaders, sans passer par les fonctions glMatrixMode et autres. Il n'est donc pas nécessaire de les initialiser.

Pour le dessin 2D, on utilise les méthodes drawText et drawPixmap pour afficher une image et un texte. Pour terminer, il faut appeler la méthode end pour indiquer que nous en avons fini avec le dessin 2D, sous peine d'avoir des bogues graphiques.

```
    QPainter painter(this);
    painter.drawText(width() - 260, 20, "Heightmap :");
    painter.drawPixmap(width() - 260, 25, 100, 100, QPixmap(
":/heightmaps/secondlife.jpg"));
    painter.drawText(width() - 130, 20, "Texture :");
    painter.drawPixmap(width() - 130, 25, 100, 100, QPixmap(
":/textures/side1.png"));
    painter.end();
```

```
}
```

Il faut noter que cet exemple n'est pas du tout optimisé. En effet, le chargement des images dans les QPixmap avec la fonction drawPixmap est effectué à chaque mise à jour du rendu. Pour optimiser, il faudrait charger l'image une seule fois puis la stocker.

OpenGL, Qt