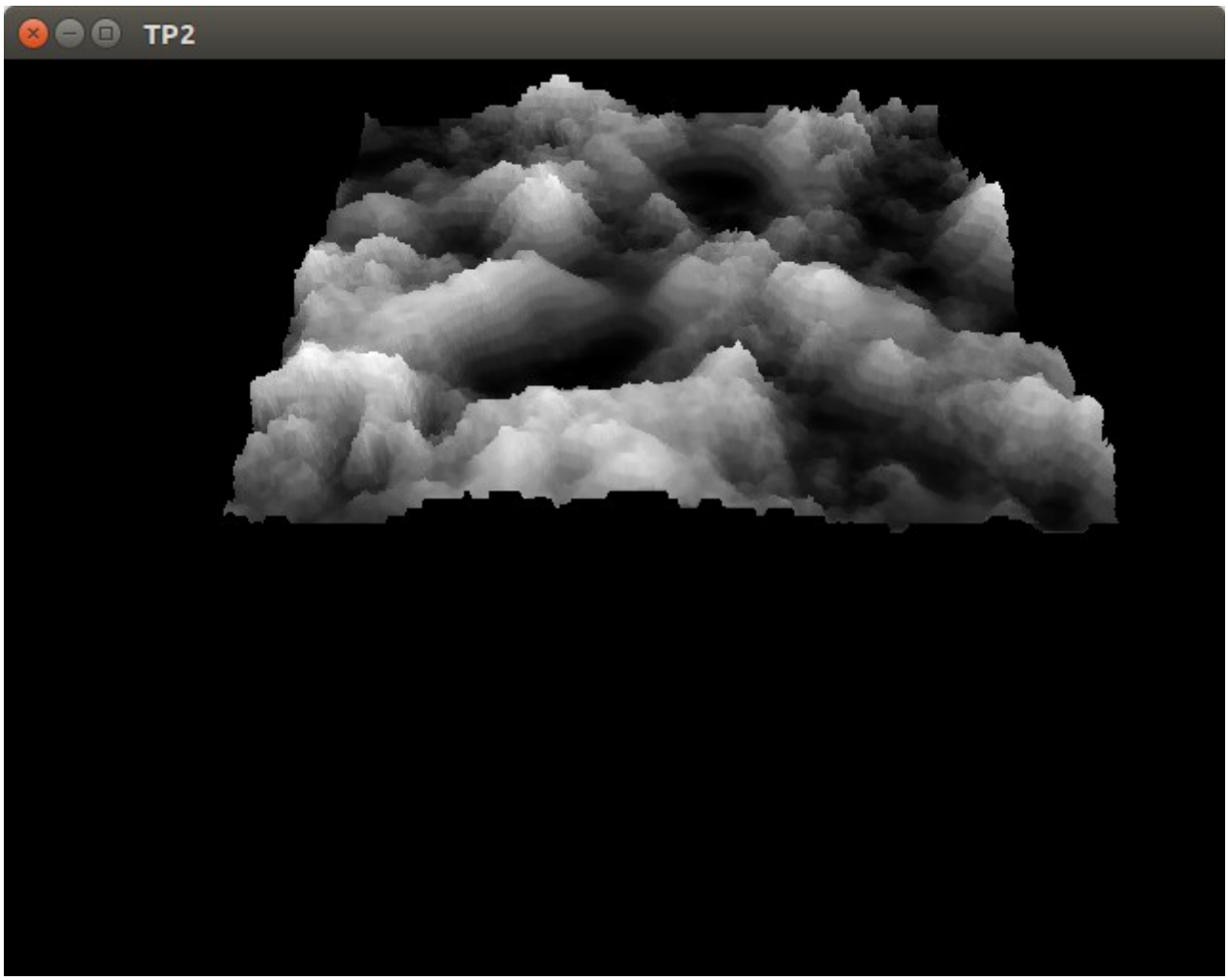
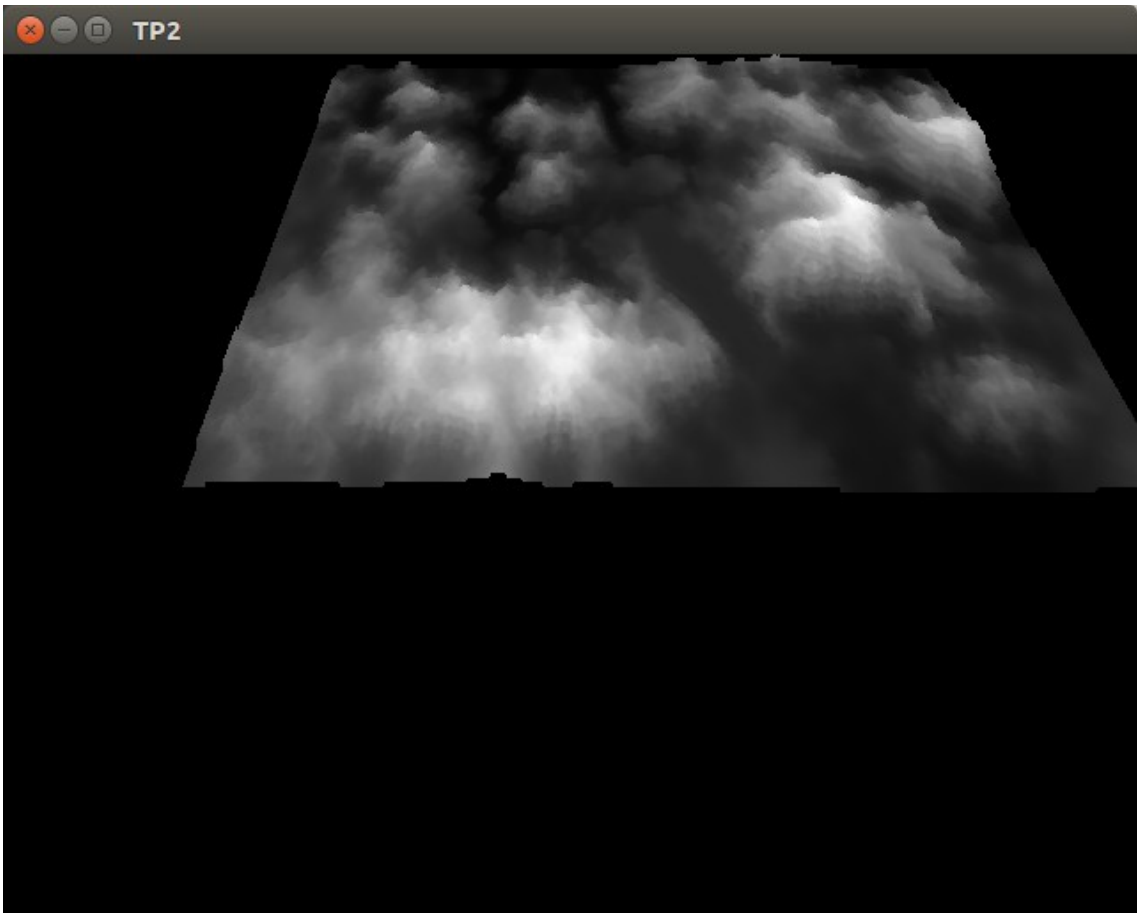
Compte Rendu TP2
Cossin Tristan

Question 1 :

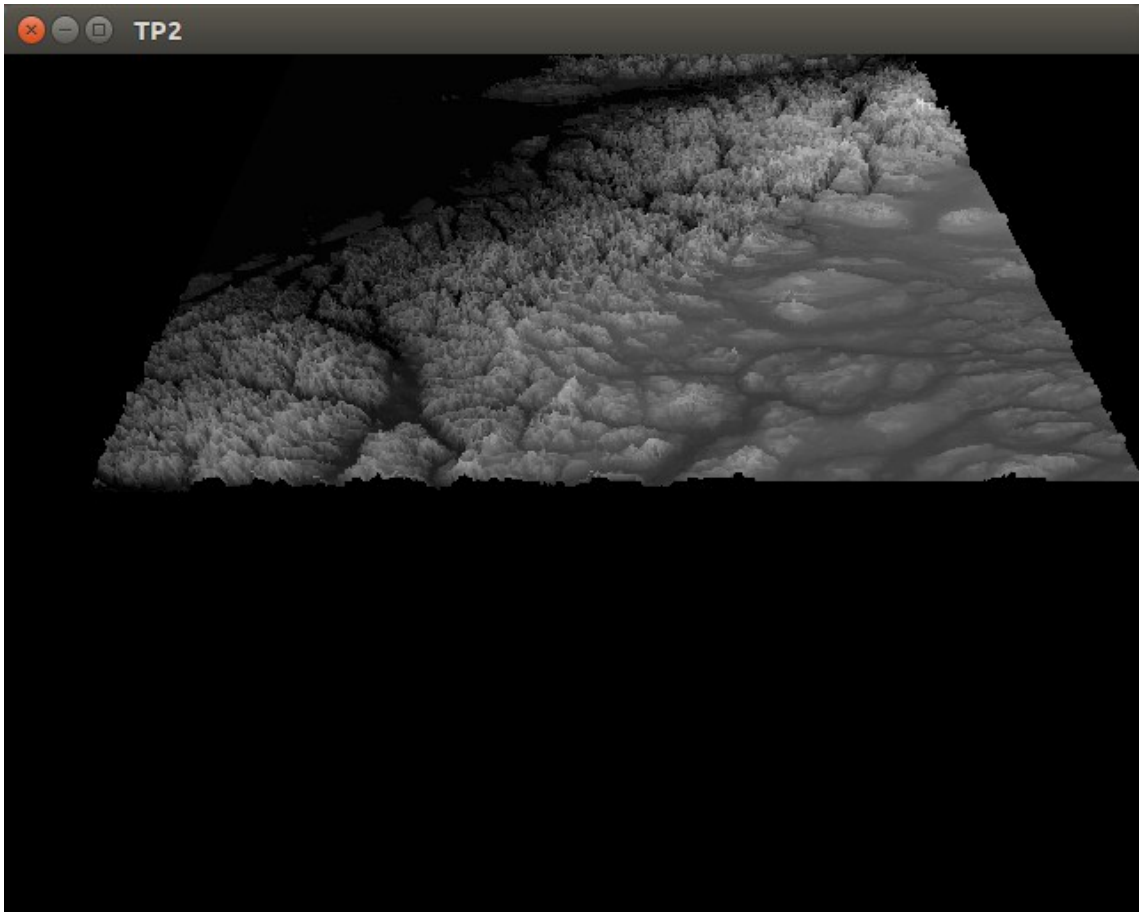
heightmap-1



heightmap-2



heightmap-3



Question 2 :

-Terrain sous un angle de 45 degré :

Pour cette partie, j'ai eu un peu de mal a comprendre l'approche et j'ai du demandé de l'aide à l'un de mes camarades qui m'a montré comment il avais fait. Son approche n'a pas été de modifié l'angle de la caméra comme je pensais faire mais d'incliné le terrain.

```
QQuaternion framing = QQuaternion::fromAxisAndAngle(QVector3D(1,0,0), -45.0);
matrix.rotate(framing);
matrix.rotate(rotation);
```

-Tourner la caméra à vitesse constante autour d'un axe :

Pour réussir, il fallait commencer par annulé la décélération de la caméra qui se trouve dans « timerEvent » puis ensuite on lui applique un axe autour de lequel tourner.

```
rotation = QQuaternion::fromAxisAndAngle(QVector3D(0.0, 0.0, 1.0), Vitesse) * rotation;
```

Question 3 :

-La MAJ du Terrain :

Le nombre de rafraîchissement par seconde et défini dans « initializeGL » puis c'est la fonction « timerEvent » qui se charge du rafraîchissement de la fenêtre .

-La classe Qtimer :

La classe Qtimer sert a fournir des timers pour pouvoir effectuer des actions a des temps donné comme le rafraîchissement de la fenêtre par exemple.

-Modification de la classe MainWideget :

Pour répondre à cette question, j'ai simplement ajouté un paramètre au constructeur qui prend en paramètre le nombre de rafraîchissement souhaiter pour la fenêtre.

```
explicit MainWidget(QWidget *parent, int frame);
```

Puis dans initializeGL :

```
timer.start(Framerate, this);
```

-Affichage de quatre fenêtres avec des fréquences différentes :

```
MainWidget widget1(nullptr, 1);
MainWidget widget10(nullptr, 10);
MainWidget widget30(nullptr, 30);
MainWidget widget100(nullptr, 100);
widget1.show();
widget10.show();
widget30.show();
widget100.show();
```

On peut observer que plus le framerate est élevé et plus l'image semble « fluide » tandis que la fenêtre à 1 image seconde semble de pas avancé.

-Flèche UP / Down

Pour l'implémentation des flèches, j'ai préféré utilisé les touche + et – car cela est plus facile à retenir et probablement durant un autre TP, ces touches seront attribuer à une autre fonction. J'ai utilisé la même manières que pour le TP1 pour l'implémentation.

```
case Qt::Key_Minus:
    Pression[6] = true;
    break;
case Qt::Key_Plus:
    Pression[7] = true;
    break;
```

Puis dans timerEvent:

```
 if (Pression[6])
 {
      if(Vitesse > 0)
         Vitesse -= 0.01;
  }
  if (Pression[7])
       Vitesse += 0.01;
```