Forum: Python & Plugins

Topic: Import fichier DEM

Subject: Import fichier DEM

PostÃ© par: Bobibou

Contribution le : 17/2/2013 16:30:53

Bonjour à tous !

On m'a demandé il y a peu un moyen d'ouvrir les fichiers DEM présents sur un CD VistaPro.
Le format DEM, pour Digital Elevation Model est un format décrivant le relief d'une portion de terrain.
En gros, c'est une heightmap mais elle n'est pas compresser de la même façon qu'une image donc je n'ai pas trouvé de moyen simple de l'ouvrir.
Avec l'aide de ce site, j'ai fait un petit script permettant d'ouvrir ces fichiers. Je n'ai testé le script que sur les fichiers du cd que j'avais mais je sais que le format DEM a été utilisé par l'USGS, United States Geological Survey, donc ça peut peut-être servir à d'autres.

En ce qui concerne l'utilisation du script, il faut indiquer dans rep le chemin du fichier à ouvrir et la variable pos permet de gérer le décalage. Par exemple, si on charge deux fichiers représentant des zones contiguës selon l'axe X, on charge le premier fichier avec pos = (0,0) et le second avec pos = (1,0). Ça évite de déplacer les blocs à la main si on veut en charger beaucoup.

Un petit rendu d'un terrain chargé avec ce script (région des Alpes italiennes) :

Voici donc le petit script d'import : profitez-en, c'est libre d'être réutilisé sans conditions de ma part.

```
import io from struct import unpack, calcsize import bpy  ## Paramètres ##     # Fichier rep = "/media/DEM_ALPES/Alps/ALPS75/DEMS0303/ALPSAR19.DEM"     # Position du bloc. # Aide : Si pour le fichier ALPSaa11, pos = (i, j) #  si aa augmente, augmenter i #  si 11 augmente, augmenter j pos = (0, 0) scale = .1 scalez = 1./500. ## ##  def main(rep, pos, scale) :     offx = pos[1] * 256 * scale     offy = pos[0] * 256 * scale         ## Ouverture du fichier     file = io.FileIO(rep)     raw = file.read()     f = &#039;B&#039;          # Format : unsigned char     s = calcsize(f)   # Taille en bytes d&#039;un élément de données     off = 2048        # Offset (2048 bytes d&#039;en-tete)         wx = 258     width = 258         ## Analyse des données     verts = []     faces = []         print("----------------")     def read(off) :         if off+s > len(raw) :             print("erreur")         (a,) = unpack(f, raw[off:off+s])         return a, off + s         x = 0     while x < wx :         # longueur de la ligne         long, off = read(off)         a, off = read(off)         long = long * 256 + a               # Décompression         k = off         data = []         while off - k < long :             c, off = read(off)             if c >= 128 :                 r = 257 - c # Nombre de répétitions                 oct, off = read(off)                 data.extend([oct]*r)             else :                 for i in range(c + 1) :                     oct, off = read(off)                     data.append(oct)                 # Altitude absolue         alt = data[0] * 256 + data[1]         point = [alt]         k = 2         while k < len(data) :             c = data[k]             if c == 128 and k + 2 < len(data) :                 point.append(256 * data[k+1] + data[k+2])                 k += 3             elif c < 128 :                 point.append(point[-1]+c)                 k +=
```

```
1 else : point.append(point[-1]-256+c) k += 1 # Construction du maillage for y in range(min(width,len(point))) : hgt = point[y] verts.append((offy + y*scale, offx + x*scale, hgt*scalez)) if x > 0 and y > 0 : faces.append(((x-1)*width + y, (x-1)*width + y - 1, x*width + y - 1, x*width + y)) x += 1 ## Construction de l&#039;objet mesh = bpy.data.meshes.new("dem_mesh") mesh.from_pydata(verts, [], faces) mesh.update() obj = bpy.data.objects.new("dem", mesh) scene = bpy.context.scene scene.objects.link(obj) main(rep, pos, scale)
```