Université Grenoble Alpes **Polytech Grenoble**
RICM5 **Année 2016–2017**

Visualisation — TP n° 2 : Création de visualisations dans Google Earth

1 Introduction

1.1 Google Earth

Google Earth permet de visualiser la terre, l'espace et plus récemment la planète Mars et la Lune, grâce à des données provenant de diverses sources (images satellites, données géologiques, images de la NASA...). Dans ce TP, nous allons apprendre à l'utiliser pour créer des visualisations portant sur la géographie.

1.2 Le format KML

Il est possible d'ajouter des éléments (images, géométrie...) dans Google Earth en utilisant un format spécial, le KML, basé sur XML. Le contenu minimal d'un fichier KML est le suivant :

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
  <Document>
    <name>Mon objet KML</name>
  </Document>
</kml>
```

Lorsque vous placez ce contenu dans un fichier portant l'extension kml (par exemple fichier.kml) et que vous l'ouvrez dans Google Earth, vous voyez apparaitre une icône qui lui est associée dans le panneau de gauche, comme dans la figure suivante :

Vous pouvez ainsi importer plusieurs fichiers KML et les activer ou désactiver. Pour bien visualiser votre contenu créé, désactivez tous les autres contenus de Google Earth en déselectionnant “Base de données primaire” en bas à gauche. Vous devez obtenir un affichage de la Terre vide comme dans la figure précédente.

Vous pouvez trouver plus d'informations sur le KML en ligne sur les sites suivants :

- `https://developers.google.com/kml/documentation/kml_tut` : tutoriel sur le kml
- `https://developers.google.com/kml/documentation/kmlreference` : documentation du format KML

1.3 Génération de fichiers kml

L'intérêt principal du KML est qu'il peut être généré par un programme externe à partir de données. Il est ainsi possible de créer des visualisations pour Google Earth selon une source de données. Pour cela, il suffit d'écrire le contenu KML dans un fichier en ajoutant les informations dépendantes des données.

Par exemple, le programme Python suivant génère un fichier KML contenant un point placé aléatoirement sur la carte :

```
 #!/usr/bin/python3

import random

latitude = random.randrange(-90, 90)
longitude = random.randrange(-180, 180)

kml = (
    '<?xml version="1.0" encoding="UTF-8"?>'
    '<kml xmlns="http://www.opengis.net/kml/2.2">'
    '<Placemark>'
    '<name>Random Placemark</name>'
    '<Point>'
    '<coordinates>
    ''')

# ajout des coordonnees aleatoires
kml += str(longitude) + "," + str(latitude)

kml += (
     '</coordinates>'
    '</Point>'
    '</Placemark>'
    '</kml>')

# affichage sur la sortie standard
print(kml)
```

Ce programme produit le contenu suivant :

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
<Placemark>
<name>Random Placemark</name>
<Point>
<coordinates>-22,70</coordinates>
</Point>
</Placemark>
</kml>
```

Ce programme Python permet d'obtenir un fichier KML pour afficher le point en redirigeant la sortie standard :

```
python3 randomMark.py > mark.kml
```

Vous n'êtes pas obligés d'utiliser Python pour ce TP, mais cet exemple montre qu'il est facile de générer du KML sans bibliothèque additionnelle ou gestion de fichiers.

2 Images

La description KML permet de positionner des images sur le sol dans Google Earth. Pour cela, il faut ajouter dans le fichier KML un noeud de type "GroundOverlay" :

```
        <GroundOverlay>
        <name>Exemple d'Overlay</name>
        <Icon>
          <href>image.jpg</href>
        </Icon>
        <LatLonBox>
          <north>45.0</north>
          <south>45.1</south>
          <east>5.6</east>
          <west>5.7</west>
        </LatLonBox>"
        <color>a0ffffff</color>
      </GroundOverlay>
```

Le chemin de l'image à placer est spécifié dans le noeud `Icon`. Il est également possible de placer un lien web pour récupérer une image en ligne.

Le noeud `LatLonBox` décrit les coordonnées correspondant aux limites de l'image sur la carte. Ces coordonnées sont spécifiées en latitude nord, latitude sud, longitude ouest et longitude est.

Les deux premiers caractères du noeud `color` (a0) permettent d'ajuster la transparence de l'image, `00` rendant l'image entièrement transparente et `ff` entièrement opaque.

Exercice 1 : Créez un fichier kml permettant de placer l'ancienne carte de la France (carte.jpg) au bon endroit sur la carte. Les coordonnées sont les suivantes :

- Longitude Nord : 51.14582064171803
- Longitude Sud : 42.22993231551517
- Latitude Ouest : -6.151180487455577
- Latitude Est : 10.79288010970161

Une fois l'image placée, il est possible d'en modifier la transparence dans Google Earth (clic droit sur l'icône du fichier → propriétés).

Exercice 2 : Trouvez les coordonnées permettant de placer le plan du campus de Grenoble fourni (PlanCampus.jpg) au bon endroit, et créez le fichier kml correspondant.

Procédez en plusieurs étapes :

1. Trouvez les lieux correspondant aux limites de la boîte englobante dans Google Earth
2. Ajoutez un repère sur ces lieux (menu ajouter → repère)
3. Une fois le repère ajouté, accédez à ses coordonnées dans la fenêtre ouverte
4. Convertissez les données angulaires en données longitude / latitude (par exemple avec le site http ://www.coordonnees-gps.fr/conversion-coordonnees-gps)

3 Positions

Dans la suite du TP, nous allons nous intéresser à la population de certaines capitales européennes. Le fichier *cities.txt* contient les lignes suivantes :

```
Paris 2.3522219000000177 48.856614 2.244
Bruxelles 4.351710300000036 50.8503396 0.177
Berlin 13.404953999999975 52.52000659999999 3.502
Rome 12.496365500000024 41.9027835 2.627
Lisbonne -9.139336599999979 38.7222524 0.530
Madrid -3.7037901999999576 40.4167754 3.165
Londres -0.12775829999998223 51.5073509 8.674
```

Chaque ligne contient dans l'ordre le nom de la ville, sa latitude, sa longitude et sa population en millions d'habitants. Nous voudrions visualiser la population de ces capitales sur la carte. Pour cela, nous allons utiliser des noeuds de type `Placemark` permettant de placer un point sur la carte :

```
<Placemark>
<name>Mon point</name>
<Point>
<coordinates>-22,70</coordinates>
</Point>
</Placemark>
```

Les coordonnées sont spécifiées sous forme (latitude, longitude). Pour placer plusieurs points, il suffit de les placer les uns à la suite des autres dans le noeud `Document` du fichier KML.

Exercice 3 : Ecrivez un programme qui lit la description des villes et qui génère un fichier KML contenant des marqueurs placés sur les villes. Vous devriez obtenir un résultat similaire à la figure suivante :

4 Animation

Il est possible de créer une animation en ajoutant des temps dans certains noeuds KML. Ces temps sont placées dans un noeud `begin`, lui-même placé dans un noeud `TimeSpan`. L'exemple suivant contient une animation faisant apparaitre deux points l'un après l'autre, les temps correspondant à des années :

```
<Placemark>
<name>Placemark 1</name>
<TimeSpan>
<begin>2000</begin>
</TimeSpan>
<Point>
<coordinates>-22,70</coordinates>
</Point>
</Placemark>

<Placemark>
<name>Placemark 2</name>
<TimeSpan>
<begin>2001</begin>
</TimeSpan>
<Point>
<coordinates>-24,70</coordinates>
</Point>
</Placemark>
```

Lorsque Google Earth lit un fichier contenant des noeuds annotés, il crée automatiquement un widget permettant de controler l'animation.

Exercice 4 : En se basant sur la question précédente, écrivez un programme permettant d'obtenir une animation dans laquelle les villes apparaissent progressivement.

5 Chemins

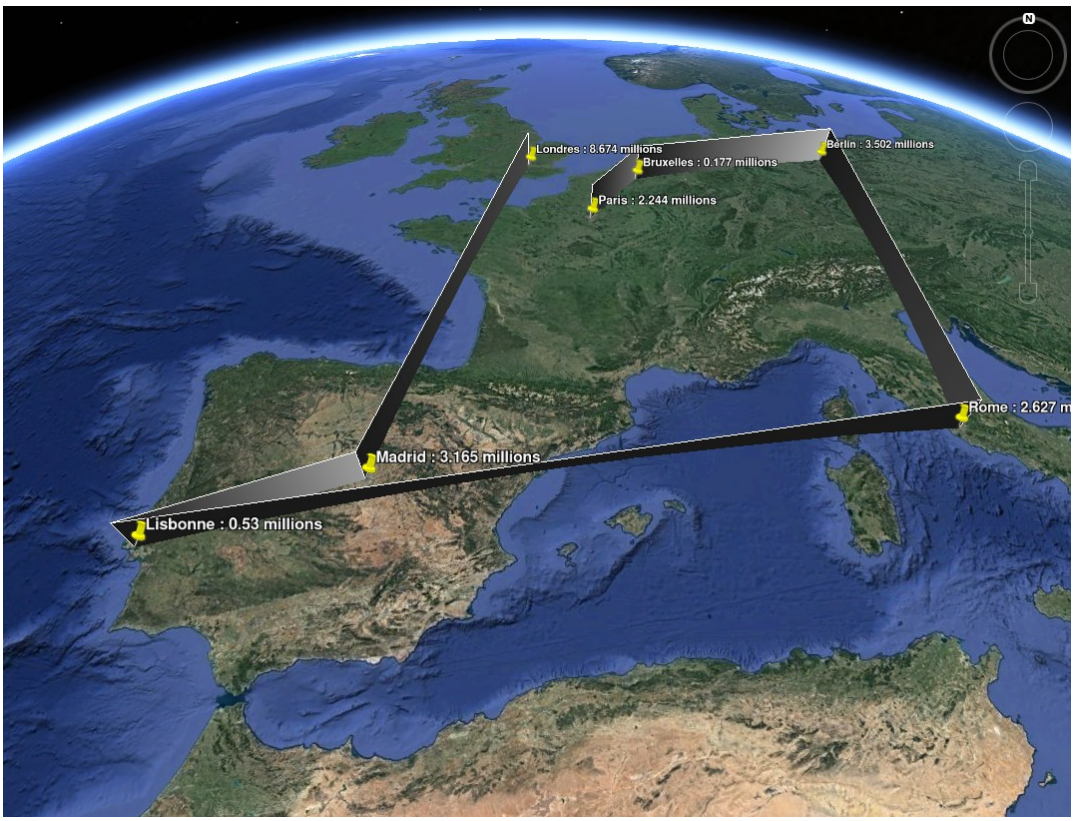
Il est possible d'ajouter des chemins grâce au noeud `LineString`, placé dans un noeud `Placemark`. L'exemple suivant affiche un chemin sur la carte :

```
<Placemark>
  <name>Absolute Extruded</name>
  <description>Transparent green wall with yellow outlines</description>
  <styleUrl>#yellowLineGreenPoly</styleUrl>
  <LineString>
    <extrude>1</extrude>
    <tessellate>1</tessellate>
    <altitudeMode>absolute</altitudeMode>
    <coordinates>
          -112.2550785337791,36.07954952145647,2357
        -112.2549277039738,36.08117083492122,2357
        -112.2552505069063,36.08260761307279,2357
        -112.2564540158376,36.08395660588506,2357
        -112.2580238976449,36.08511401044813,2357
        -112.2595218489022,36.08584355239394,2357
        -112.2608216347552,36.08612634548589,2357
        -112.262073428656,36.08626019085147,2357
        -112.2633204928495,36.08621519860091,2357
        -112.2644963846444,36.08627897945274,2357
        -112.2656969554589,36.08649599090644,2357
    </coordinates>
  </LineString>
</Placemark>
```

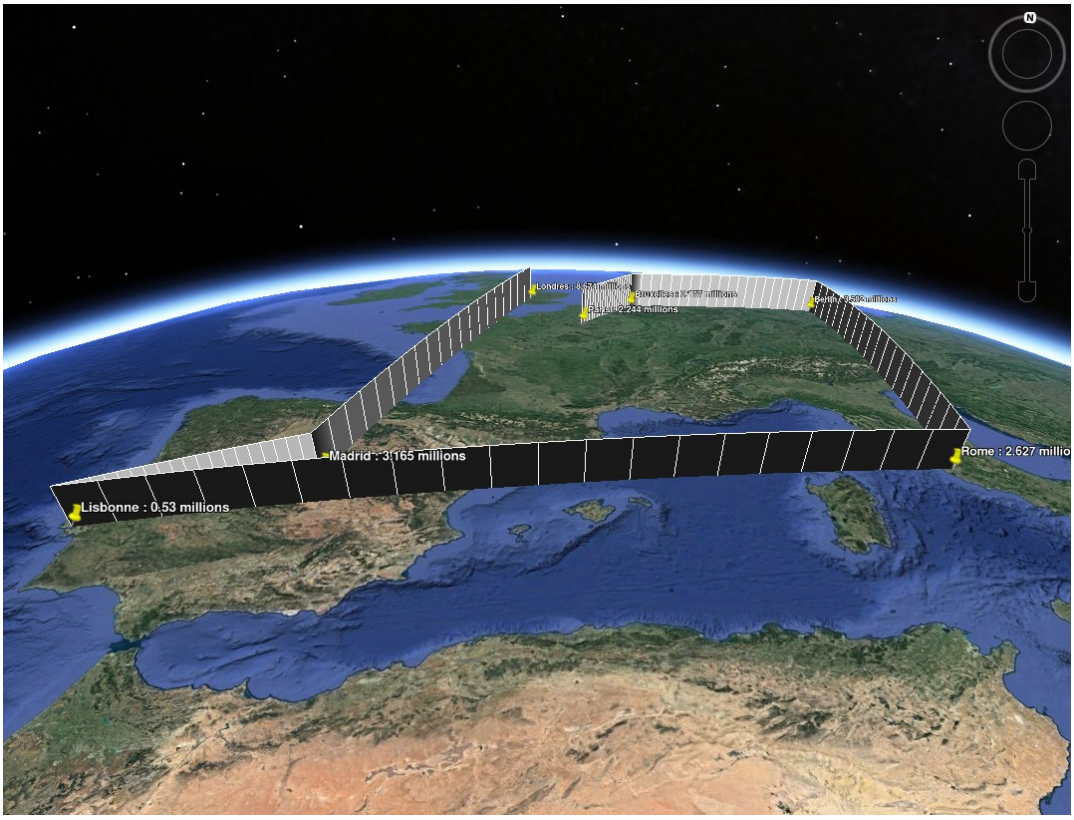
Vous remarquez que les points ont 3 coordonnées. Il est en effet possible de spécifier l'altitude des points, ce qui peut permettre de voir le chemin de loin.

Exercice 5 : En se basant sur la question précédente, écrivez un programme permettant d'afficher un chemin entre les villes.

Vous pouvez constater que le chemin créé passe parfois sous la terre, comme dans la figure suivante :



Pour éviter cela, il faut ajouter des points intermédiaires pour que le chemin suive la courbure de la terre, comme dans la figure suivante :



Exercice 6 : Ecrivez un programme générant un chemin entre les villes, en générant un certain nombre de points intermédiaires.

Exercice 7 : Ecrivez un programme permettant de tracer ce chemin progressivement au cours d'une animation.

6 Polygones

Il est possible d'afficher des polygones sur la carte comme dans l'exemple suivant :

```
<Placemark>
    <name>The Pentagon</name>
    <Polygon>
      <extrude>1</extrude>
      <altitudeMode>relativeToGround</altitudeMode>
      <outerBoundaryIs>
        <LinearRing>
          <coordinates>
            -77.05788457660967,38.87253259892824,100
            -77.05465973756702,38.87291016281703,100
            -77.05315536854791,38.87053267794386,100
            -77.05552622493516,38.868757801256,100
            -77.05844056290393,38.86996206506943,100
            -77.05788457660967,38.87253259892824,100
          </coordinates>
        </LinearRing>
      </outerBoundaryIs>
      <innerBoundaryIs>
        <LinearRing>
          <coordinates>
            -77.05668055019126,38.87154239798456,100
            -77.05542625960818,38.87167890344077,100
            -77.05485125901024,38.87076535397792,100
            -77.05577677433152,38.87008686581446,100
            -77.05691162017543,38.87054446963351,100
            -77.05668055019126,38.87154239798456,100
          </coordinates>
        </LinearRing>
      </innerBoundaryIs>
    </Polygon>
  </Placemark>
```

Cet exemple (source : Google) permet d'afficher un polygone représentant le Pentagone. Un polygone est défini par sa bordure extérieure, suite de points placée dans un noeud `outerBoundaryIs`, et sa bordure intérieure, définie dans le noeud `innerBoundaryIs`.

Exercice 8 : Ecrivez un programme lisant le fichier des villes et générant un fichier kml contenant des polygones placés sur les villes. Vous pouvez utiliser de simples polygones, par exemple des cubes définis par 4 points pour la bordure extérieure.

Exercice 9 : Ecrivez un programme pour que la hauteur des polygones soit proportionnelle à la population des villes.

Voici un exemple de résultat :

Exercice 10 : Ecrivez un programme pour que la largeur des polygones soit proportionnelle à la population des villes.

Voici un exemple de résultat :

7 Style

Jusqu'à maintenant, toutes nos constructions géométriques étaient sans couleur. Cela va changer grâce à la définition des styles. Les styles peuvent être définis dans un noeud correspondant à un objet géométrique :

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
  <Document>
    <name>Exemple de style</name>
    <Placemark>
      <name>Path avec style</name>
        <Style>
        <LineStyle>
                <color>7f00ffff</color>
                <width>4</width>
        </LineStyle>
        <PolyStyle>
                <color>7f00ff00</color>
        </PolyStyle>
        </Style>
      <LineString>
        <extrude>1</extrude>
        <tessellate>1</tessellate>
        <altitudeMode>absolute</altitudeMode>
        <coordinates>
           -112.2550785337791,36.07954952145647,2357
           -112.2549277039738,36.08117083492122,2357
           -112.2552505069063,36.08260761307279,2357
        </coordinates>
      </LineString>
    </Placemark>
  </Document>
</kml>
```

Dans cet exemple, on définit le style des lignes du chemin dans le noeud `LineStyle`, et le style des surfaces entre les lignes dans le noeud `PolyStyle`.

Il est également possible de définir le style en début de fichier pour pouvoir le réutiliser dans plusieurs objets géométriques. Pour cela, il faut lui associer un identifiant et y référer via le noeud `styleUrl` dans les objets géométriques. L'exemple suivant illustre ce fonctionnement :

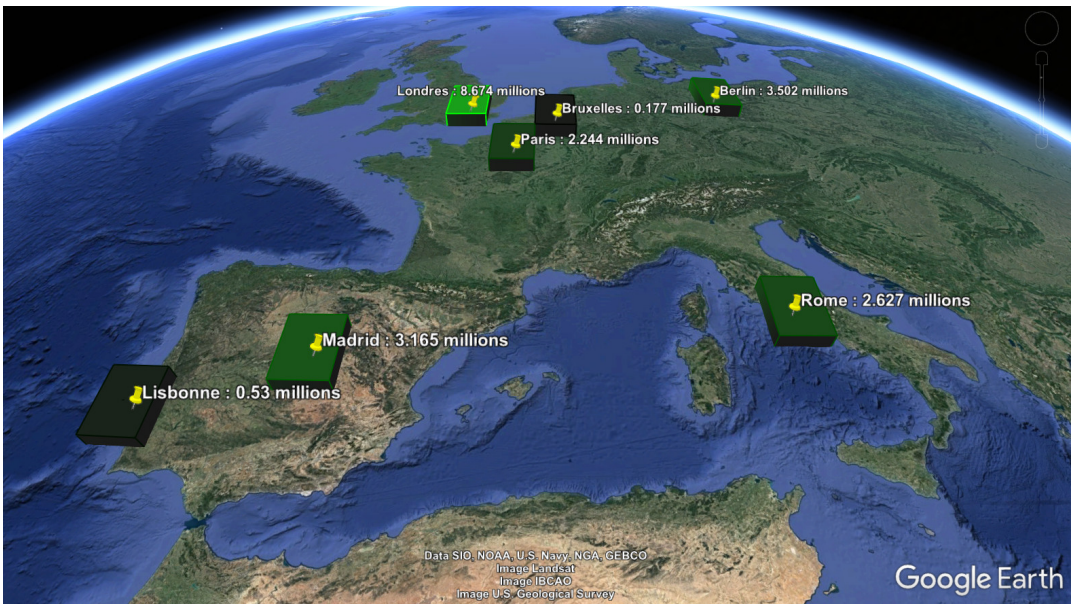
```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
  <Document>
    <name>Exemple de style</name>
    <Style id='yellowLineGreenPoly'>
      <LineStyle>
        <color>b000ffff</color>
        <width>4</width>
      </LineStyle>
      <PolyStyle>
        <color>b000ffff</color>
      </PolyStyle>
    </Style>
    <Placemark>
      <name>Path avec style</name>
      <styleUrl>#yellowLineGreenPoly</styleUrl>
      <LineString>
        <extrude>1</extrude>
        <tessellate>1</tessellate>
        <altitudeMode>absolute</altitudeMode>
        <coordinates>
          -112.2550785337791,36.07954952145647,2357
          -112.2549277039738,36.08117083492122,2357
          -112.2552505069063,36.08260761307279,2357
          -112.2564540158376,36.08395660588506,2357
        </coordinates>
      </LineString>
    </Placemark>
  </Document>
</kml>
```

Exercice 11 : Ecrivez un programme se basant sur la question 6 pour appliquer un style au chemin tracé. Variez la taille, la transparence et la couleur du chemin.

Exercice 12 : Ecrivez deux programmes se basant sur les questions 9 et 10, où les polygones sont affichés dans une couleur de votre choix.

Exercice 13 : Ecrivez un programme se basant sur la question précédente, où la couleur des polygones est associée à la population des villes.

Voici un exemple de résultat :


Londres : 8.674 millions
Berlin : 3.502 millions
Bruxelles : 0.177 millions
Paris : 2.244 millions
Rome : 2.627 millions
Madrid : 3.165 millions
Lisbonne : 0.53 millions
Data SIO, NOAA, U.S. Navy, NGA, GEBCO
Image Landsat
Image IBCAO
Image U.S. Geological Survey
Google Earth