5) Tracé de voies

5.1) Tracé de voies de chemin de fer

5.1.1) Requêter les données *Open Street Map*

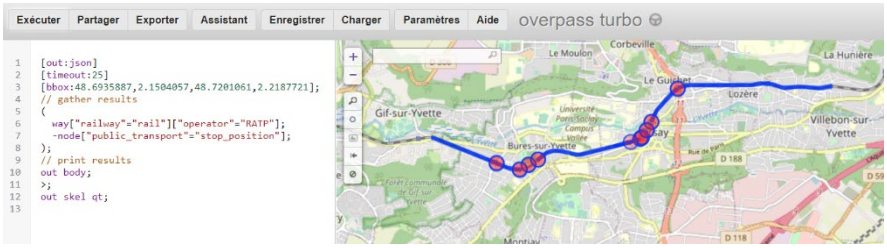
OpenStreetMap est la carte ouverte et collaborative du Monde, améliorée chaque jour par plus d'un million de contributeurs.

Vous pouvez requêter les données via *Overpass turbo*, l'interface web de filtrage de données d'*OpenStreetMap* : https://overpass-turbo.eu/. Cette interface permet d'exécuter des requêtes de l'*API Overpass* et d'analyser les données d'*OpenStreetMap* interactivement sur une carte.

Entrez votre première requête en langage *Overpass QL*[11] pour obtenir le tracé du tronçon de la ligne du RER B qui traverse votre terrain :

```
[out:json]
[timeout:25]
[bbox:48.6935887,2.1504057,48.7201061,2.2187721];
// gather results
(
  way["railway"="rail"]["operator"="RATP"];
);
// print results
out body;
>;
out skel qt;
```

Cliquez sur « Exécuter » pour avoir la prévisualisation de vos données :



5.1.2) Exploiter les données *Open Street Map*

Pour récupérer vos données, cliquez sur « Exporter » puis choisissez d'exporter les « Données » via le lien « Télécharger/copier en tant que GeoJSON ».

```
{
  "type": "FeatureCollection",
  "generator": "overpass-ide",
  "copyright": "The data included in this document is from www.openstreetmap.org. The
data is made available under ODbL.",
  "timestamp": "2021-02-05T21:47:03Z",
  "features": [
    {
```

[11] La documentation du langage est accessible via : Wiki Open Street Map – Overpass QL

```
      "type": "Feature",
      "properties": {
        "@id": "way/4746303",
        "electrified": "contact_line",
        "frequency": "0",
        "gauge": "1435",
        "importance": "regional",
        "name": "Ligne de Sceaux (RER B4)",
        "operator": "RATP",
        "railway": "rail",
        "short_name": "RER B4",
        "start_date": "1854-07-29",
        "usage": "main",
        "voltage": "1500"
      },
      "geometry": {
        "type": "LineString",
        "coordinates": [
          [
            2.1841371,
            48.6980952
          ],
          [
            2.1842441,
            48.6981338
          ],
…
```


Ce fichier est à la norme *GeoJSON*, il est composé d'une *FeatureCollection* qui est un tableau de *Feature*. Comme dans votre tracé GPS précédent, chaque *Feature* comprend deux parties : Un sous objet *geometry* qui décrit les données spatiales, et un sous objet *properties* qui contient les informations associées.

La géométrie de vos voies ferrées comprend plusieurs poly-lignes (il s'agit de polygones non fermés) de type *LineString*. Le tableau *coordinates* liste les points successifs, eux-mêmes représentés dans un tableau à deux dimensions contenant la longitude et la latitude du point.

5.1.3) Intégrer les données *Open Street Map*

Construisez une nouvelle classe *Railways* pour gérer les voies ferrées. Le constructeur de cette classe appelé par *this.railways = new Railways(this.map, "railways.geojson");* construira une *PShape railways* de type *GROUP*, qui sera constituée des formes individuelles des poly-lignes récupérées.

Pour chaque poly-ligne, vous procèderez comme pour votre tracé GPS précédent, en utilisant les objets *GeoPoint* et *ObjectPoint* de la classe *Map3D*. Bien que la requête spécifie une boite englobante correspondant à votre terrain, *OpenStreetMap* renvoie certains points externes à cette boite pour les segments qui sont « à cheval » sur l'aire spécifiée. Comme vous ne disposez pas de données d'élévations hors de votre terrain vous devrez, à défaut d'un calcul de *clipping* plus élaboré, exclure ces points soit en utilisant la méthode *GeoPoint.inside()* qui renvoie *false*

si un point est situé hors de la carte, soit en vérifiant que l'élévation renvoyée n'est pas négative.

Vous ajouterez à votre classe les méthodes *update()* et *toggle()* pour gérer l'affichage et la visibilité des voies ferrées, que vous affecterez à la touche « *R* » du clavier.

5.1.4) Transformation de tracés en surfaces

Quand vous créez une poly-ligne constituée de points successifs, Processing traite cette forme dans un espace « assimilé » à deux dimensions : Les tracés sont orientés face à la caméra et leur taille, spécifiée par *strokeWeight*, reste constante indépendamment de la distance de la caméra.

Pour obtenir une forme définie dans un espace à trois dimensions, vous devez transformer vos tracés en surfaces planes, ce qui revient à transformer vos poly-lignes en *PShape* de type *QUAD_STRIP*.

Tout d'abord, considérez une route droite constituée d'un départ A et d'une arrivée B :

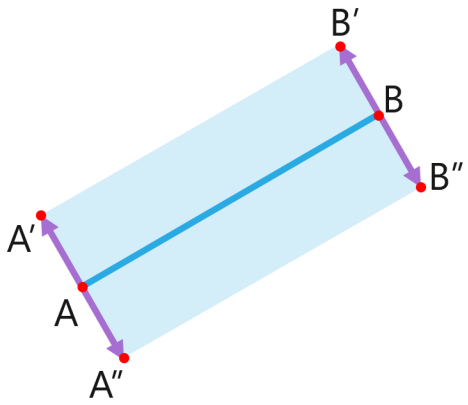


Figure 18 - Route droite de A à B

Pour élaborer la forme *QUAD_STRIP* A'A''B'B'', calculez le vecteur *Va* orthogonal à AB. Une fois Va normalisé, vous le multipliez par *laneWidth/2* pour tracer une route de la largeur *laneWidth*. Vous ajoutez ou soustrayez le vecteur *Va* à A pour obtenir A' et A'', puis ajoutez ou soustrayez le vecteur *Va* à B pour obtenir B' et B''.

La classe *PVector* de Processing dispose des méthodes nécessaires à de tels calculs. Par exemple, si A & B sont des *PVector*, vous pouvez écrire :

```
PVector Va = new PVector(A.y - B.y, B.x - A.x).normalize().mult(laneWidth/2.0f);
lane.normal(0.0f, 0.0f, 1.0f);
lane.vertex(A.x - Va.x, A.y - Va.y, A.z);
lane.normal(0.0f, 0.0f, 1.0f);
lane.vertex(A.x + Va.x, A.y + Va.y, A.z);
```

La voie étant plane[12], les élévations sont identiques à celles récupérées lors de la création des *ObjectPoint*.

Considérez maintenant une route « sinueuse » constituée d'un départ A et d'une arrivée C qui passe par un point B intermédiaire :

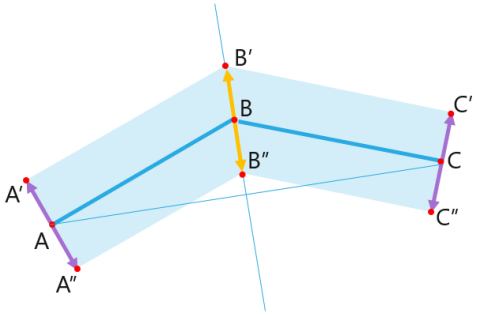


Figure 19 - Route sinueuse de A à C

Pour élaborer la forme *QUAD_STRIP* A'A''B'B''C'C'' :

- Démarrez par A'A'' calculés comme précédemment orthogonalement à AB.
- Pour tous les points intermédiaires du tracé, comme ici B, calculez le vecteur *Vb* orthogonal à AC. Une fois *Vb* normalisé, vous le multipliez par *laneWidth/2* pour tracer une route de la largeur *laneWidth*, puis vous ajoutez ou soustrayez *Vb* à B pour obtenir B' et B''.
- Terminez votre *QUAD_STRIP* par C'C'' calculés comme précédemment orthogonalement à BC.

Félicitations, vous avez tracé une première voie ferrée à partir de données d'*OpenStreetMap* :

Figure 20 - Tracé de la ligne de RER B

5.1.5) Décalage des élévations

Dans le cas de segments rectilignes, vous serez amenés à calculer des tracés entre des points

[12] S'agissant de surfaces planes, vous pouvez forcer pour chaque sommet une normale en (0,0,1)

relativement distants, qui peuvent de ce fait être sécants à des parties du terrain.

Votre tracé apparaîtra correctement avec un terrain en affichage « fil de fer »[13], mais avec des zones manquantes en affichage utilisant la texture d'image satellite :

Figure 21 - Tracés sécants au terrain

Dans le cas d'un terrain accidenté, il serait nécessaire d'ajouter des points intermédiaires dans les tracés, afin que leurs dénivelés « collent » au terrain (l'IGN fournit des élévations avec une précision de 5 mètres).

Dans ce projet, le terrain n'est pas trop accidenté, aussi vous pouvez utiliser un artifice visuel qui consiste simplement à rehausser les élévations des tracés par rapport au sol.

Par exemple, vous pouvez rehausser artificiellement les rails de 7,5 mètres avant la transformation du tracé en polygones :

```
Map3D.GeoPoint gp = this.map.new GeoPoint(point.getDouble(0), point.getDouble(1));
if (gp.inside()) {
  gp.elevation += 7.5d;
  Map3D.ObjectPoint op = this.map.new ObjectPoint(gp);
  path.add(op.toVector());
}
```

5.2) Tracé de routes

Vous venez d'achever une étape laborieuse. La bonne nouvelle, c'est que vous pouvez désormais généraliser votre travail à tous types de tracés de voies (routes, voies fluviales, lignes de bus, etc.).

5.2.1) Requêter les données *Open Street Map*

Pour tracer les routes, vous utiliserez les données fournies issues de la requête en langage *Overpass QL* suivante :

```
[out:json][timeout:25][bbox:48.6935887,2.1504057,48.7201061,2.2187721];
```

[13] Le tracé apparaitra également correctement sur une vue aplatie, en forçant *mode3D* à *false* dans la classe Map3D.

```
(
way
  ["highway"]
  ["area"!="yes"];
);
// print results
out body;
>;
out skel qt;
```

Cette fois-ci vous aller extraire tous les tracés (*way*) de type « route » (*highway*).

5.2.2) Exploiter les données *Open Street Map*

La construction des tracés sera similaire à votre voie ferrée.

Dupliquez votre classe *Railways* pour créer votre nouvelle classe *Roads* pour gérer les routes. Le constructeur de cette classe appelé par *this.roads = new Roads(this.map, "roads.geojson");* construira une *PShape roads* de type *GROUP*, qui sera constituée des formes individuelles des poly-lignes récupérées.

Vous améliorerez votre traitement pour utiliser le type de voie trouvé dans la propriété « *highway* » afin de déterminer la taille, la couleur et le décalage d'élévation des différents types de routes.

Voici un exemple de catégorisation possible des routes, des plus importantes aux moins importantes, dont les couleurs[14] sont inspirées des cartes officielles *OpenStreetMap* :

```
String laneKind = "unclassified";
color laneColor = 0xFFFF0000;
double laneOffset = 1.50d;
float laneWidth = 0.5f;
// See https://wiki.openstreetmap.org/wiki/Key:highway
laneKind = properties.getString("highway", "unclassified");
switch (laneKind) {
case "motorway":
  laneColor = 0xFFe990a0;
  laneOffset = 3.75d;
  laneWidth = 8.0f;
  break;
case "trunk":
  laneColor = 0xFFfbb29a;
  laneOffset = 3.60d;
  laneWidth = 7.0f;
  break;
case "trunk_link":
case "primary":
  laneColor = 0xFFfdd7a1;
  laneOffset = 3.45d;
  laneWidth = 6.0f;
  break;
case "secondary":
case "primary_link":
```

[14] Voir : Wiki OpenStreetMap - Highway

```
  laneColor = 0xFFf6fabb;
  laneOffset = 3.30d;
  laneWidth = 5.0f;
  break;
case "tertiary":
case "secondary_link":
  laneColor = 0xFFE2E5A9;
  laneOffset = 3.15d;
  laneWidth = 4.0f;
  break;
case "tertiary_link":
case "residential":
case "construction":
case "living_street":
  laneColor = 0xFFB2B485;
  laneOffset = 3.00d;
  laneWidth = 3.5f;
  break;
case "corridor":
case "cycleway":
case "footway":
case "path":
case "pedestrian":
case "service":
case "steps":
case "track":
case "unclassified":
  laneColor = 0xFFcee8B9;
  laneOffset = 2.85d;
  laneWidth = 1.0f;
  break;
default:
  laneColor = 0xFFFF0000;
  laneOffset = 1.50d;
  laneWidth = 0.5f;
  println("WARNING: Roads kind not handled : ", laneKind);
  break;
}

// Display threshold (increase  if more performance needed...)
if (laneWidth < 1.0f)
  break;
```

5.2.3) Transformation de tracés en surfaces

Le traitement est identique à celui des voies ferrées.

Félicitations, vous pouvez désormais circuler sur les nombreuses voies de votre terrain :

Figure 22 - Tracé des routes