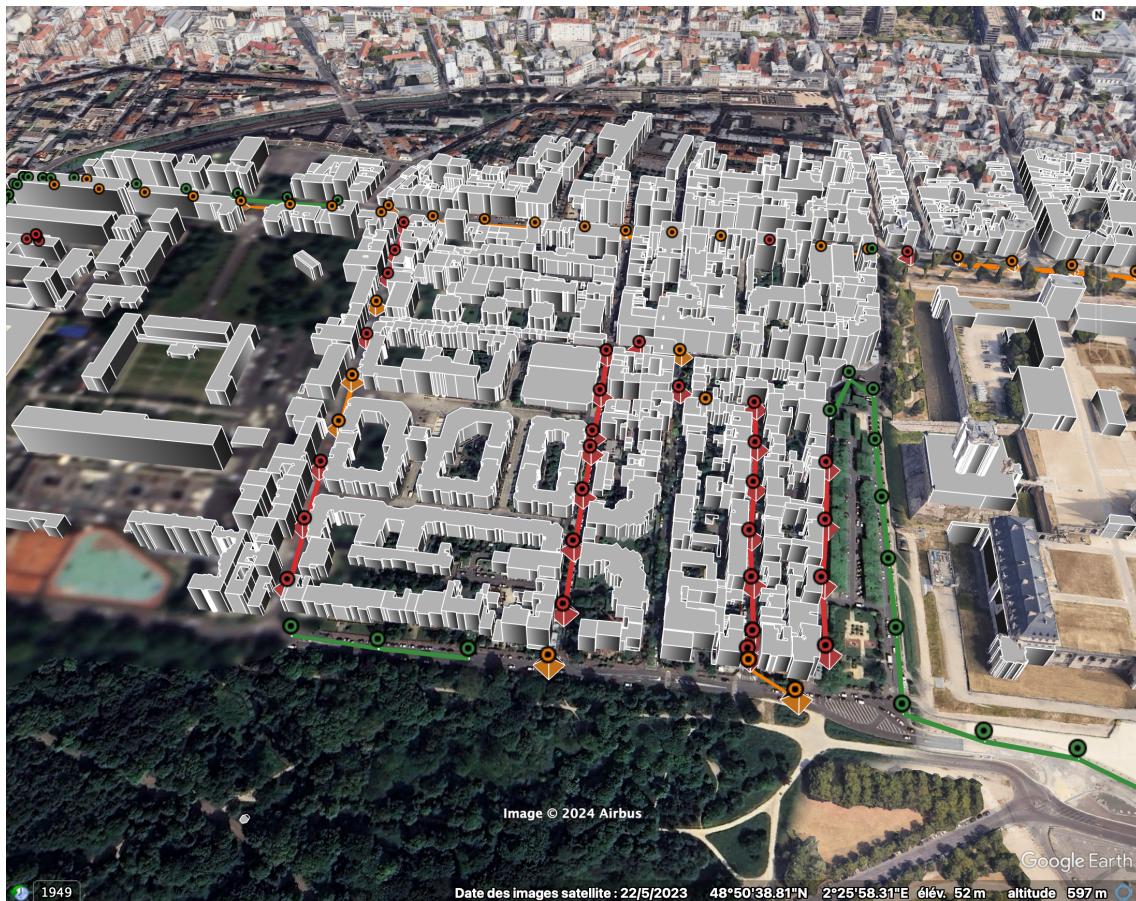


Année Universitaire 2023 – 2024
Unité de Recherche : ENSG
Encadrant(s) : Mehdi Daakir

Rapport technique - PDI 17

Gabin Bourlon, Axel De Bock, Clément Cambours & Félix Mercier

Semestre d'hiver 2023



Sommaire

1	Introduction	3
1.1	Source du projet	3
1.2	Contexte du projet	3
1.3	Glossaire	3
2	Besoins et Objectifs du projet	3
2.1	Les besoins	3
2.2	Les objectifs	4
3	Méthodologie de travail	4
4	Solution technique	5
4.1	Fonctionnalités	5
5	Bilan du projet	11
5.1	Conclusion	11
5.2	Perspectives	11
6	Références	12

1 Introduction

1.1 Source du projet

Le code développé et libre de droit. Il est disponible sur github via :

<https://github.com/Geobsys/csv2kml-qc/tree/dev>

Pour l'instant toujours en phase de développement, notre travail est localisé sur la branche dev du répertoire.

1.2 Contexte du projet

L'étude que nous avons réalisée est commanditée par Mehdir Daakir, membre de l'équipe "**Acquisition et Traitement des Données**" au sein du laboratoire de recherche LASTIG (Laboratoire en Sciences et Technologies de l'Information Géographique) de l'IGN (Institut National de l'Information Géographique et Forestière). Dans le cadre de la recherche menée au sein du LASTIG, il est important de disposer d'outils efficaces pour évaluer la qualité des données de trajectoire GNSS. Les données GNSS sont utilisées dans divers contextes de recherche géospatiale, tels que la cartographie, la géodésie, la navigation et la modélisation environnementale. Cette initiative s'inscrit dans un contexte où l'équipe de recherche cherche à améliorer ses outils d'analyse et de traitement des données GNSS. La vérification de la qualité des trajectoires GNSS est une étape cruciale dans de nombreuses études spatiales, et il est nécessaire de disposer d'outils efficaces pour effectuer cette vérification de manière rapide et simple. Par conséquent, la création de cet outil de préparation à la visualisation de trajectoire GNSS répond directement à cette problématique en offrant une solution pratique pour évaluer la qualité des données de trajectoire. En convertissant les fichiers CSV en un format KML pour une visualisation cartographique (notamment à l'aide de visualiseurs comme Google Earth), cet outil permettra aux chercheurs du LASTIG de mieux appréhender et d'analyser les données de trajectoire GNSS dans le cadre de leurs recherches en géomatique et en analyse spatiale. L'enjeu est de réussir à proposer une solution simple et efficace pour l'analyse de qualité de trajectoire GNSS.

Ce projet s'inscrit dans un module de notre enseignement, et pour lequel est dédié une journée par semaine pendant trois mois.

1.3 Glossaire

CSV : Comma-separated values

Les fichiers CSV (CSV) sont des fichiers texte (format texte ouvert), qui permettent d'enregistrer un tableau de valeur en séparant les colonnes par des virgules.

KML : Keyhole Markup Language

Le KML est un langage fondé sur le formalisme du XML (Extensible Markup Language). Ce langage permet de décrire des objets géométriques, et est ensuite interprété par des visualiseurs comme QGIS ou Google Earth. Les données sont enregistrées dans un fichier KML.

2 Besoins et Objectifs du projet

2.1 Les besoins

L'outil csv2kml doit proposer une visualisation à travers un fichier KML de la qualité d'une acquisition GNSS à partir d'un simple fichier CSV contenant une liste de points mesurés par un récepteur GNSS

avec des attributs supplémentaires. La qualité d'une acquisition étant ici synonyme d'une bonne précision sur la mesure d'un point. Cette visualisation doit permettre d'afficher de nombreux attributs relatifs aux points de l'acquisition tels que la vitesse, l'orientation, les coordonnées du capteur, le statut de la position (*RTK float, RTK fixed, DGNSS, None*) ou encore l'heure de l'acquisition. Les bâtiments alentours doivent être modélisés en 3D. En outre, l'outil doit aussi modéliser l'orientation du récepteur obtenu grâce à une centrale INS MEMS ainsi que les satellites observés lors de la prise d'une mesure.

2.1.1 Les contraintes

L'outil doit se présenter sous forme de code Python et se lancer à l'aide de la ligne de commande. Un fichier CSV contenant des mesures de points est nécessaire au bon fonctionnement de l'outil. Des fichiers supplémentaires tels qu'un fichier shapefile des bâtiments de la zone d'étude ou un fichier Rinex associé à la prise de mesure permettent d'obtenir des informations et fonctionnalités supplémentaires. L'outil doit se servir de ressources Open source. Il existe deux formats de mesure différents, 'LOG' et 'EXTEVENT'. Le format 'LOG' contient les données de journalisation GNSS proprement dites. Il enregistre les mesures GNSS capturées par le récepteur à intervalles réguliers.

Le format 'EXTEVENT' contient des informations sur les événements externes qui se produisent pendant l'enregistrement des données GNSS au moment de la mesure. Il dispose d'informations supplémentaires telles que des valeurs angulaires (utiles pour l'orientation).

L'outil doit être capable de traiter les deux formats. Pour cela, l'argument "it" (ou "input_type") permet à l'utilisateur de spécifier le type de fichier inséré.

2.2 Les objectifs

L'objectif de ce projet est de créer un outil simple d'utilisation, comportant 6 fonctionnalités principales :

- F.1 Affichage des attributs associés à chaque position : la date, l'heure, le statut de la position, la latitude (en WGS84), la longitude (en WGS84), la hauteur (en mètres), les incertitudes planimétriques (en mètres), l'incertitude d'altimétrie, la position en RGF93 (X,Y,Z), l'altitude, la distance depuis la mesure précédente, le temps depuis la mesure précédente, le temps depuis la première mesure, le temps restant avant la dernière mesure, la vitesse, le nombre de satellites visibles par le récepteur lors de la mesure, les noms des satellites visibles.
- F.2 Affichage de la trajectoire segmentée selon un certain critère (exemple : segments fixés / non fixés, ...),
- F.3 Affichage des ellipses/ellipsoïdes de confiance associées à chaque position,
- F.4 En contexte urbain, un affichage des emprises en 3D des bâtiments pour évaluer la visibilité des satellites,
- F.5 Affichage des directions récepteur – satellites à partir de la trajectoire et des éphémérides,
- F.6 Affichage de l'orientation du récepteur via les valeurs angulaires mesurées.

3 Méthodologie de travail

Après avoir analysé les différentes tâches à réaliser, nous les avons classées selon leur difficulté. Par conséquent, nous avons réalisé les différentes fonctionnalités dans l'ordre : de F.1 à F.4 puis F.6 et F.5. Les conditions de travail ne changent que très peu entre chaque fonctionnalité. Le développement

se fait entièrement en Python à l'aide de bibliothèques spécifiques en prenant en paramètre le fichier de mesures. Les fonctionnalités F.4 et F.5, doivent être complétées par d'autres fichiers. Pour F.4 (Bâtiments), il est nécessaire d'avoir une couche Shapefile comprenant les bâtiments dans la zone environnante en paramètre. Un exemple de ce type de fichier est la BD TOPO de l'IGN, qui fournit l'ensemble des bâtiments sur un département. Il suffit ainsi de posséder le fichier du département correspondant à la zone d'étude. Par ailleurs, pour la fonctionnalité F.5, les fichiers Rinex associés aux mesures sont nécessaires. Ceux-ci se trouvent directement dans le récepteur GNSS après l'acquisition. Il contient des informations sur les satellites captés par le récepteur pendant l'acquisition.

4 Solution technique

Chacune des fonctionnalités nécessitent des calculs et des ressources de différents types, qui nous ont parfois amenés à importer des bibliothèques python. Pour gérer les données de chaque point, nous avons fait le choix d'utiliser les bibliothèques *pandas* et *numpy*, qui permettent une gestion sous forme de *DataFrame*, et des calculs en bloc sur l'ensemble des points.

En parallèle, de nombreux calculs nécessitent une conversion de coordonnées dans des repères géographiques, cartésiens ou autres. Pour cela, nous avons choisi d'utiliser la bibliothèque *pyproj* qui permet de réaliser ces calculs de manière fiable.

Dans le processus de création de fichiers KML, nous avons intégré la bibliothèque *simplekml*. Elle permet de créer des fichiers, d'intégrer des objets et de paramétriser l'affichage final.

Ensuite, d'autres bibliothèques sont spécifiques à certaines fonctionnalités, comme *fiona* et *shapely* (F.4) ou *gnsstoolbox* et *gpsdatetime* (F.5).

4.1 Fonctionnalités

4.1.1 F.1 Affichage d'attributs associés à chaque position

Les fichiers 'LOG' ou 'EXTEVENT' possèdent des attributs associés à chaque point. Le format choisi dans cette étude reste expérimental et est voué à s'affiner. En revanche, ils permettent tous deux d'accéder à la position en WGS84, à son incertitude et à son mode de calcul (*RTK float*, *RTK fixed*, *DGNSS*, *None*), et enfin à la date et heure de prise de mesure.

Aux informations de positionnement s'ajoutent des informations sur l'orientation de l'appareil dans les fichiers 'EXTEVENT'.

Cette fonctionnalité permet simplement de rendre ces informations visibles lors de la visualisation du fichier une fois transformé au format KML. Il est possible de sélectionner un point en cliquant dessus pour voir apparaître une fenêtre recueillant tous ces attributs. D'autres attributs consultables dans cette même fenêtre ont été calculés : c'est le cas de la vitesse du capteur, du temps écoulé depuis le début de l'acquisition ou de la distance avec le point précédent. Pour finir, cette fonction permet de mettre à jour les descriptions des points avec les éléments calculés par les autres fonctionnalités.

Certains attributs conditionnent les paramètres d'affichage des points : par exemple, la couleur du point est donnée par l'état de l'acquisition (vert, orange, rouge et gris) :

- Le vert correspond au statut de position RTK fixed. Associé à une mesure de bonne précision (cm).
- L'orange correspond au RTK float, dont la précision est moins bonne (dm).
- Le rouge est associé au statut DGNSS, qui traduit une mauvaise précision (m).

- Le gris signifie qu'il n'y a pas de donnée (erreur de mesure, perte de connexion...).

L'implémentation de cette fonctionnalité passe par une fonction permettant de générer une description (Figure 1 ci-dessous) à partir d'un point. En créant une fonction spécialisée dans la génération de descriptions, on développe une architecture qui sera facilement transposable aux autres éléments à décrire.

Index	5520/14728
Unknown	1
GNSS	GNSS
Unknown	2
Time	133h 53min 37.1s
Date	23/02/2024
Hour	13:53:37.100
State	RTK_Fix
Latitude (WGS84)	48.84103012 °
Longitude (WGS84)	2.433565412 °
Height	95.263 m
Uncertainties Lat-Lon	0.03 m
Uncertainty High	0.07 m
X (RGF93)	4194076.012 m
Y (RGF93)	4797782.476 m
Z (RGF93)	269014.502 m
Altitude	51.495 m
Distance since prev pt	68.058 m
Time laps since prev pt	0h 0min 10.0s
Time last since first pt	0h 9min 10.0s
Time left	0h 15min 20.8s
Velocity	6.071 m/s

FIGURE 1 : Affichage des attributs

4.1.2 F.2 Affichage de la trajectoire segmentée

Cette fonctionnalité permet d'afficher un segment reliant les points un à un pour former un aperçu de la trajectoire réelle (Figure 2 ci-dessous). Cet affichage est paramétré par l'état d'acquisition (*RTK float*, *RTK fixed*, *DGNSS*, *None*), et produit des lignes reliant les points de même état avec une couleur donnée.

De la même manière que les points, chaque segment est indexé et porte une description permettant de le situer dans le relevé (1er et dernier point, et statut de mesure).



FIGURE 2 : Affichage des segments - visualisation sur Google Earth

4.1.3 F.3 Visualisation des incertitudes planimétriques et altimétriques associées à chaque position

Cette fonctionnalité permet de visualiser simplement les incertitudes relatives à la position du point (Figure 3). Si initialement la représentation des incertitudes devait passer par une représentation ellipsoïdale, l'absence de matrice de covariance et le format de fichier kml ne nous ont pas permis de représenter ces incertitudes sous cette forme. Nous avons pris l'initiative de représenter ces incertitudes par des pyramides à base carrée.

La base de la pyramide est déterminée par l'incertitude planimétrique et la hauteur par l'incertitude altimétrique. Pour rendre la visualisation plus facile, il est possible de définir un seuil d'incertitude (planimétrique ou altimétrique) à partir duquel la pyramide est affichée en noir, afin de distinguer au premier coup d'œil si les incertitudes sont importantes. Il est aussi possible de définir un facteur d'échelle (planimétrique ou altimétrique) pour agrandir les pyramides trop peu visibles lorsque l'incertitude est inférieure au décimètre.

Les incertitudes sont données en mètre dans le CSV et doivent être transformées en degré décimal pour l'affichage. Cette transformation peut prendre du temps, c'est pourquoi un équivalent est au préalable calculé.

Le calcul de l'équivalent passe par une série d'étapes. Tout d'abord, la fonction calcule les coordonnées moyennes des points géographiques fournis, comprenant la latitude, la longitude et l'altitude. Ces coordonnées sont calculées dans le système de projection Lambert93. Ensuite, ces coordonnées moyennes sont converties en coordonnées WGS84 à l'aide de transformations de coordonnées géographiques. Après avoir obtenu les coordonnées moyennes dans le système WGS84, la fonction détermine deux nouveaux points à une certaine distance (cette distance est un paramètre défini arbitrairement, nous avons choisi 1 kilomètre, car il permet d'obtenir un résultat satisfaisant et reste adapté à des chantiers de quelques kilomètres d'emprise) de ces coordonnées moyennes, dans les directions Est et Nord. Ces nouveaux points sont calculés en ajoutant cette distance, appelée *size*, aux coordonnées moyennes dans le système WGS84. Une fois ces points déterminés, la fonction calcule les écarts de longitude et de latitude entre

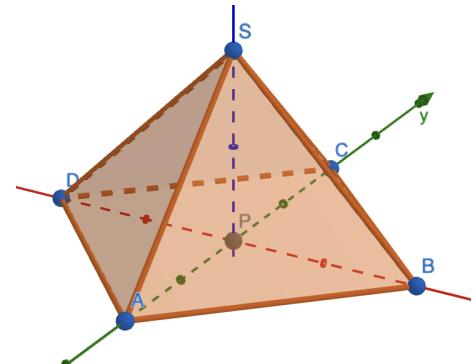


FIGURE 3 : $\|\overrightarrow{PB}\|$ l'incertitude Nord-Sud, $\|\overrightarrow{PC}\|$ l'incertitude Est-Ouest, et $\|\overrightarrow{PS}\| = h$ l'incertitude altimétrique avec P le point mesuré.

ces deux points. Ce qui permet d'obtenir un facteur d'incertitude planimétrique dans les directions Est et Nord en divisant ces écarts par la distance spécifiée *size*.

Ce ratio peut ensuite être utilisé pour ajuster les incertitudes de chaque point en vue de créer les pyramides.

Note : En KML, la pyramide est construite par 4 triangles pour visualiser les faces supérieures de la pyramide.

4.1.4 F.4 Affichage des emprises en 3D des bâtiments pour évaluer la visibilité des satellites

Cette fonctionnalité permet de visualiser l'emprise spatiale des bâtiments (Figure 4 ci-dessous). Elle permet notamment de repérer les hauts bâtiments pour expliquer une éventuelle imprécision dans l'acquisition. Calculer cette emprise permet aussi de voir si un de ces bâtiments se trouve entre le récepteur et le satellite.

Cette fonctionnalité s'utilise en plusieurs étapes. Il faut d'abord récupérer les fichiers bâtiments du département de la **BD TOPO** en v3.0 (depuis 2019), puis indiquer le chemin du fichier dans l'instruction. Ensuite, une intersection spatiale entre ces bâtiments et l'emprise du chantier (élargie par une marge paramétrable) est réalisée pour ne garder que les bâtiments environnants au chantier. L'utilisation d'une *boundary box* permet de réduire considérablement le temps nécessaire à la réalisation de cette intersection qui était réalisée bâtiment par bâtiment dans les premières versions. Cette intersection spatiale crée un fichier SHP que l'utilisateur peut sauvegarder.

Une fonctionnalité spécifique utilise ensuite chaque bâtiment sélectionné pour créer une reproduction visualisable dans le fichier KML. On peut sélectionner les attributs voulus pour les intégrer aux objets créés (comme la hauteur du bâtiment, ou son altitude). Dans le fichier KML, un bâtiment n'est pas indiqué comme un solide en trois dimensions, mais comme un polygone horizontal élevé de la hauteur du bâtiment par rapport au sol. Les côtés de ce polygone sont ensuite "étendus" jusqu'au sol par défaut. Le bâtiment est au final approché par un parallélépipède à base polygonale quelconque. De cette manière, les bâtiments sont représentés dans la visualisation.

La présence de ces objets pourrait permettre d'identifier les satellites obstrués par les bâtiments avec la fonctionnalité suivante (F.6)



FIGURE 4 : Affichage des bâtiments - visualisation sur Google Earth

4.1.5 F.5 Affichage des directions récepteur – satellites à partir de la trajectoire et des éphémérides

Dans le but d'obtenir une indication sur la précision de l'acquisition, on peut déterminer la liste des satellites visibles lors des mesures pour chaque point, et donc leur nombre.

Cette fonctionnalité utilise en entrée les Rinex d'observation et de navigation de l'acquisition pour déterminer les satellites observés. On ajoute ensuite la liste et le nombre de satellites observés dans la description de chaque point.

Par le biais de la bibliothèque *gnsstoolbox* on peut déterminer la position estimée de chaque satellite observé à l'instant de la mesure. Cela nous permettrait de définir les vecteurs reliant les points aux satellites observés.

Cette fonctionnalité devait permettre d'identifier les trajets sécant à un bâtiment, mais les calculs d'intersections se sont avérés trop complexes dans le temps imparti. En revanche, le calcul des coordonnées des satellites est tout de même réalisé dans le but d'une future extension.

4.1.6 F.6 Affichage de l'orientation du récepteur via les valeurs angulaires mesurées

Dans les fichiers de type 'EXTEVENT' sont enregistrées des mesures angulaires permettant d'orienter la photo prise dans un repère local. À partir de ces orientations, cette fonctionnalité doit pouvoir donner une visualisation de l'emprise de la photo depuis le point mesuré. Cela peut être représenté par le biais d'un frustum (Figure 6), dont le sommet est le point mesuré, d'un capteur éloigné de la distance focale (dont on définit la taille), et d'un écran de projection (défini par sa distance au point). Ces différents paramètres sont modifiables par l'utilisateur, mais sont fixés de manière arbitraire par défaut. La figure 5 ci-dessous montre le rendu de ces frustums.

La création du frustum passe par plusieurs étapes. On génère tout d'abord un frustum orienté vers le haut, dans un repère orthogonal simple à l'aide de la focale et de la distance à l'objet capturé. Il faut ensuite considérer trois matrices de rotation selon les axes X, Y et Z qui permettent d'orienter notre frustum dans le repère de l'appareil photo. Ces rotations sont définies par les mesures d'orientation du capteur ('oX', 'oY', et 'oZ' dans le CSV). Puis appliquer à nouveau 3 matrices qui permettent d'orienter la caméra dans le repère local (en l'occurrence WGS84), et qui sont elles définies par l'utilisateur ('alpha', 'beta', 'gamma').

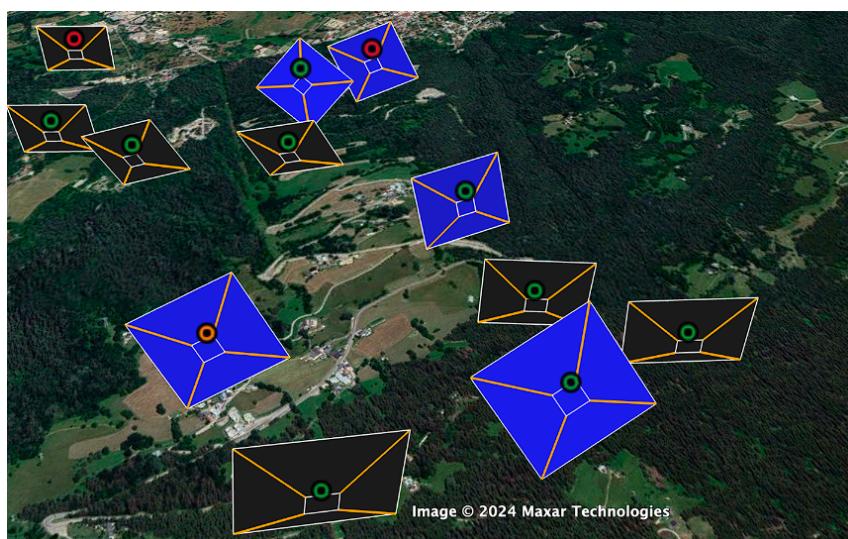


FIGURE 5 : Affichage des frustums - visualisation sur Google Earth

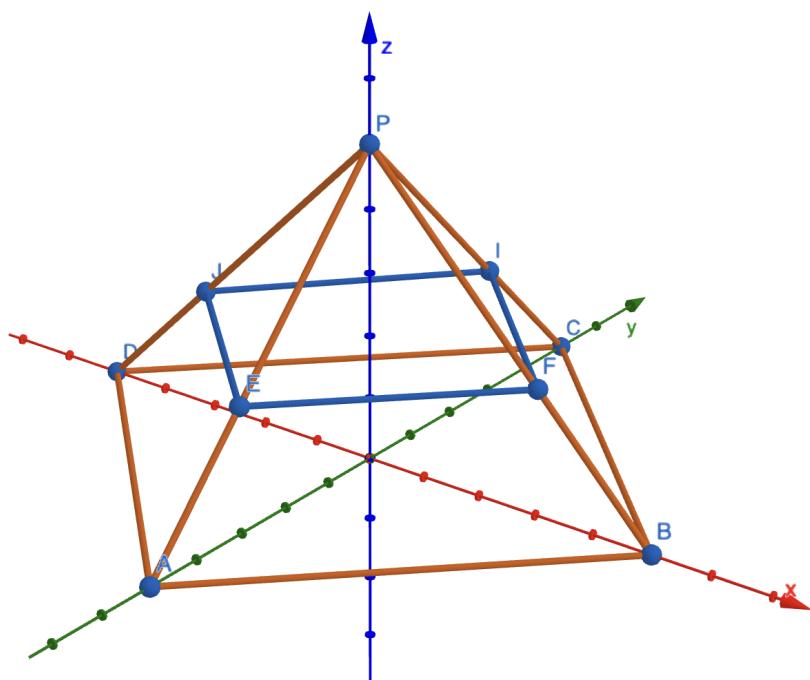


FIGURE 6 : Exemple de Frustum : *P* le point, *EFIJ* le capteur, et *ABCD* l'écran.

5 Bilan du projet

5.1 Conclusion

La problématique initiale était de pouvoir donner des attributs et des informations visuelles supplémentaires à une prise de mesures de points par récepteur GNSS. Dès lors, nous avons :

- ajouté les attributs supplémentaires aux points,
- créé des segments de couleur (nous avons créé une hiérarchisation selon la précision des mesures) entre les points afin de les relier en une trajectoire cohérente,
- transformé les ellipsoïdes de confiance en intervalles de confiance sous forme de pyramide de couleur en lien avec la précision des mesures,
- fait apparaître les bâtiments en 3D dans la zone d'étude,
- créé des frustums permettant d'obtenir une orientation de la caméra lors de la prise de mesure avec une caméra,
- et enfin, nous avons obtenu la liste des satellites captés par le récepteur lors de la mesure de chaque point.

La quasi-totalité des fonctionnalités demandées lors de ce projet a été réalisée. On notera que la piste d'affichage de l'intersection entre les satellites et les bâtiments a été écartée par manque de temps.

5.2 Perspectives

Ainsi, nous avons plusieurs pistes d'amélioration possibles pour nos résultats, telles que l'ajout d'un symbole/forme représentant l'intersection entre les bâtiments et la vue des satellites. De surcroît, nous avons envisagé le paramétrage d'une potentielle caméra : sa focale et sa taille de capteur, mais nous n'avions pas de taille précise. Il semble alors intéressant de paramétrier selon les caractéristiques d'une vraie caméra.

6 Références

Google Earth : <https://www.google.fr/intl/fr/earth/index.html>

BD TOPO : <https://geoservices.ign.fr/bdtopo>

fiona : <https://pypi.org/project/fiona/>

gnsstoolbox : <https://pypi.org/project/gnsstoolbox/>

gpsdatetime : <https://pypi.org/project/gpsdatetime/>

numpy : <https://numpy.org>

pandas : <https://pandas.pydata.org>

pyproj : <https://pypi.org/project/pyproj/>

shapely : <https://pypi.org/project/shapely/>

simplekml : <https://pypi.org/project/simplekml/>