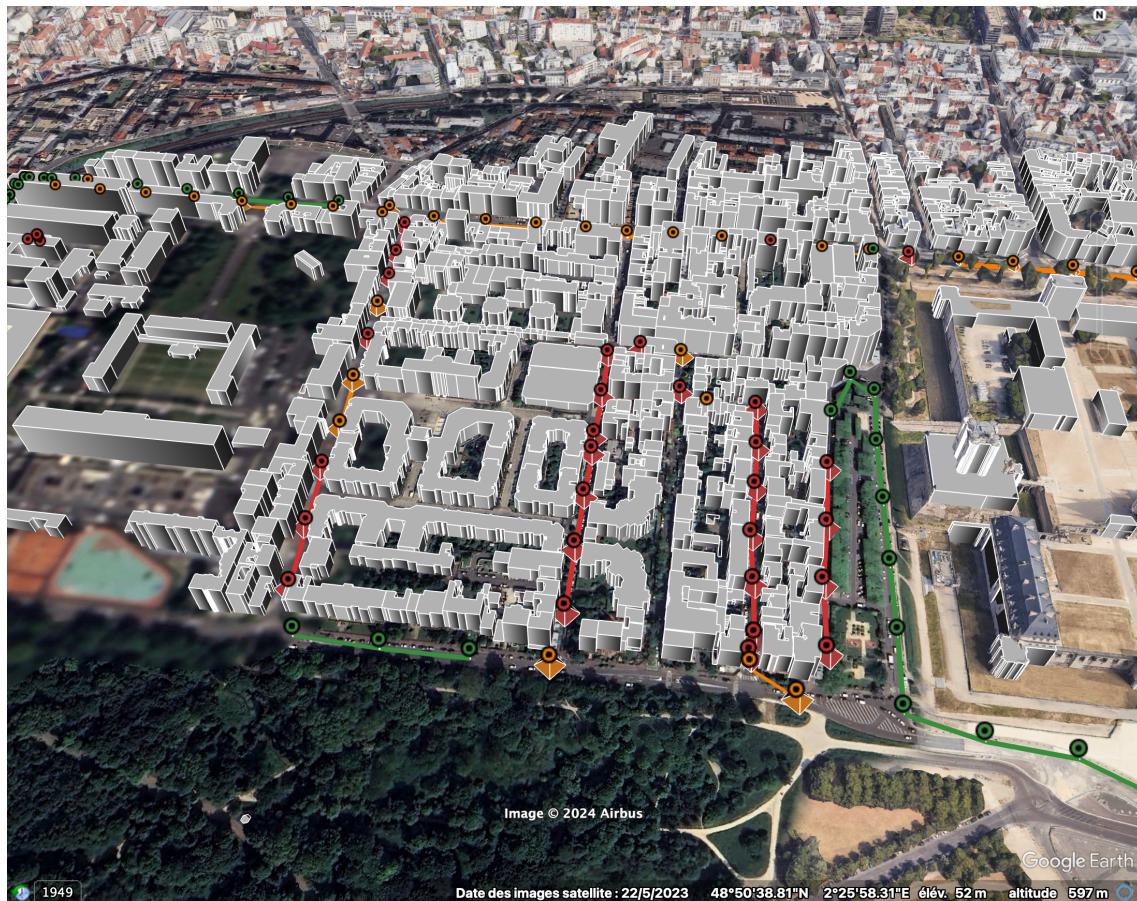


Année Universitaire 2023 – 2024
Unité de Recherche : ENSG
Encadrant(s) : Mehdi Daakir

Rapport d'analyse finale - 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	Objectifs de l'étude	4
2.1	Objectifs	4
2.2	Contraintes	4
3	Analyse fonctionnelle	6
3.1	Principe de fonctionnement	7
4	Présentation des fonctionnalités	9
4.1	F.1 Affichage des attributs associés à chaque position	9
4.2	F.2 Affichage de la trajectoire segmentée	9
4.3	F.3 Visualisation des incertitudes planimétriques et altimétriques associées à chaque position	10
4.4	F.4 Affichage des emprises en 3D des bâtiments pour évaluer la visibilité des satellites	11
4.5	F.5 Affichage des directions récepteur – satellites à partir de la trajectoire et des éphémérides	11
4.6	F.6 Affichage de l'orientation du récepteur via les valeurs angulaires mesurées	12
4.7	Outil global	13
5	Réalisation et suivi du projet	13
5.1	Risques	13
5.2	Planning suivi versus planning prévisionnel	14
6	Conclusion	16
7	Références et outils	17

1 Introduction

1.1 Source du projet

Le code développé est 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 des 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 capacités 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 tel que 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 quatre mois.

Le développement informatique de la solution se réalise sur python, à l'aide de bibliothèques publiques (simplekml, pandas, numpy...). Il n'y a donc pas de frais de licence.

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 format 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.

RTK : Real Time Kinematic

Le RTK est une technique de positionnement par satellite qui utilise des corrections en temps réel pour améliorer la précision des données GNSS, permettant ainsi des mesures de localisation très précises. Il existe différents critères de précision : - **fixed** signifie que le récepteur GNSS a pu obtenir un verrouillage

sur un signal de référence, ce qui garantit une précision élevée et une position géographique précise. - **float** signifie que le récepteur GNSS n'a pas encore obtenu de verrouillage sur un signal de référence stable, ce qui peut entraîner une précision moindre et une position géographique moins précise.

DGNSS : Differential Global Navigation Satellite System

Le DGNSS utilise des stations de référence au sol pour calculer et diffuser des corrections de signal aux récepteurs GNSS. Celui-ci possède une précision inférieure au RTK.

2 Objectifs de l'étude

2.1 Objectifs

D'un point de vue général, le projet vise à convertir des données CSV vers un fichier KML. Cette conversion doit s'accompagner d'un ensemble de traitements permettant d'apporter des informations concernant la trace GNSS relevée. Les traces GNSS dont nous disposons pour tester nos différentes implémentations ont été réalisées par notre commanditaire à l'aide d'un **GEO STIX**. Il s'agit d'un ensemble de points pris à intervalles de temps réguliers retracant une trajectoire (la trajectoire représente dans le cadre de notre projet, le chemin pris lors des mesures). Voici une liste des fonctionnalités que notre outil doit permettre de réaliser :

1. Affichage des attributs associés à chaque position.
Au minimum, la position, la vitesse (lors de la mesure), le temps écoulé depuis la première mesure, les incertitudes, le nombre de satellites et les coordonnées du point (en WGS84/Lambert93).
2. Affichage de la trajectoire segmentée selon un certain critère. Par exemple, le statut de la mesure (*RTK float, RTK fixed, DGNSS, None*).
Cela permettra d'identifier rapidement les segments dont la précision varie.
3. Visualisation des incertitudes planimétriques et altimétriques associées à chaque position.
Cet élément qualitatif permet une visualisation rapide de l'incertitude associée à un point.
4. Affichage des emprises en 3D des bâtiments pour évaluer la visibilité des satellites
Cela permet d'anticiper l'occlusion potentielle des bâtiments sur le récepteur GNSS, ou des multi-trajets.
5. Affichage des directions récepteur – satellites à partir de la trajectoire et des éphémérides
En complément de l'emprise 3D des bâtiments, ces rayons permettront d'identifier des trajets récepteur - satellites obstrués.
6. Affichage de l'orientation du récepteur via les valeurs angulaires mesurées
Cette dernière fonctionnalité permettra l'utilisation de ce programme lorsque le récepteur est couplé à un appareil photographique dans un relevé de photogrammétrie, et de visualiser les prises de vue.

2.2 Contraintes

Le code est implanté en Python à la demande du commanditaire, en utilisant des bibliothèques libres de droits. Cependant, l'affichage n'est pas pris en charge et repose sur des outils de lecture de fichiers .kml tels que Google Earth, par exemple. De plus, le lancement de l'outil se fait via des lignes de commandes. Nous utilisons GitHub pour communiquer les codes entre les membres du groupe et avec notre commanditaire.

Le développement de cet outil a été réalisé sur une période de 4 mois dédiée au projet, à raison d'une journée par semaine en général.

Dans un contexte d'utilisation, l'utilisateur doit disposer d'un fichier CSV au format correct (tel que décrit dans la documentation), de la portion de la base de données TOPO autour de la zone étudiée au format de fichier shapefile (qu'il devra télécharger au préalable en fonction du département où se trouve la zone d'étude), ainsi que des fichiers Rinex de navigation et d'observation (ces fichiers sont directement disponibles dans le récepteur GNSS après la prise des mesures). Avec ces fichiers, le programme permet de générer un fichier KML visualisable dans des logiciels tels que Google Earth.(Figure 1)

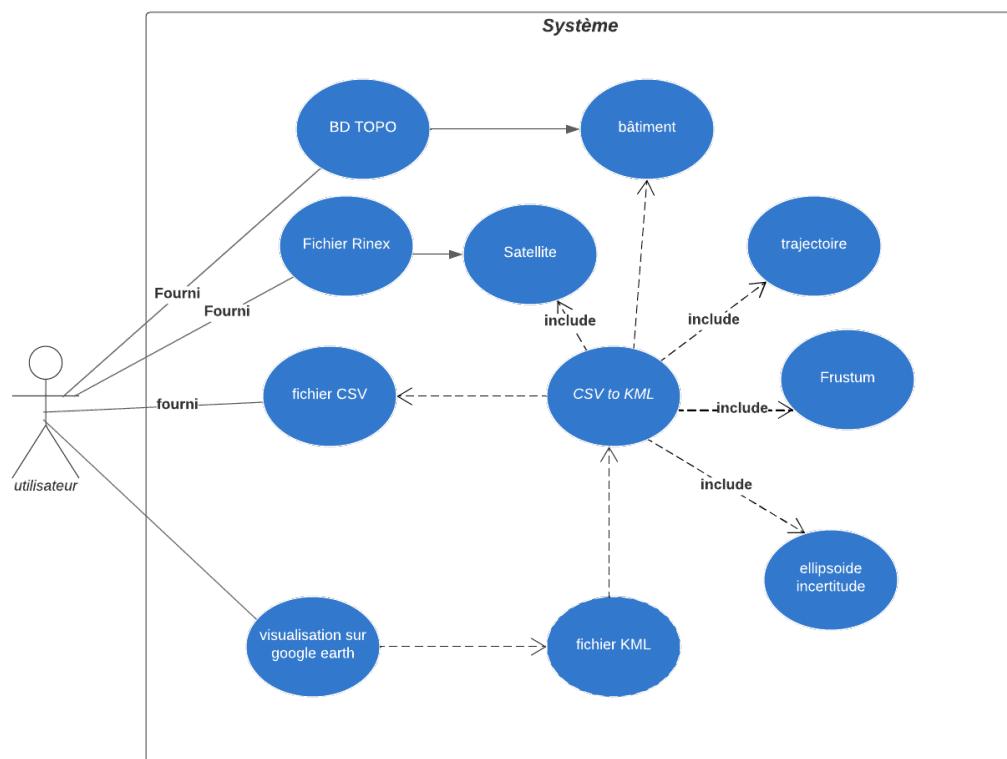


FIGURE 1 : Diagramme d'utilisation

3 Analyse fonctionnelle

Voici les fonctionnalités principales :

F1. Affichage des attributs associés à chaque position

Le but final est d'afficher chaque point à sa position, ainsi l'ensemble des points forme une approximation de la trajectoire réelle de l'acquisition. Il faut afficher une fenêtre au clic sur un des points de la trajectoire montrant les différents paramètres de l'acquisition : date, état de la position, coordonnées géographique, incertitudes, orientation et tous les autres paramètres calculés par la suite ; à ce point précis.

F2. Affichage de la trajectoire segmentée

Intégration de signes/couleurs lors de l'affichage des points de la trajectoire pour indiquer, en fonction d'un critère, une première approximation de sa précision. Par exemple, en fonction du statut de la mesure (*RTK float, RTK fixed, DGNSS, None*). Chaque segment devra donc avoir une couleur en fonction de son statut, RTK fixed = vert, RTK float = orange, DGNSS = rouge et None = gris).

F3. Visualisation des incertitudes planimétriques et altimétriques associées à chaque position

Bien que l'objectif initial fût d'afficher l'ellipsoïde de confiance, le manque de données statistiques rend le calcul de l'ellipsoïde de confiance impossible. Il faudra donc afficher les intervalles de confiance pour chaque position.

F4. Affichage des emprises en 3D des bâtiments pour évaluer la visibilité des satellites

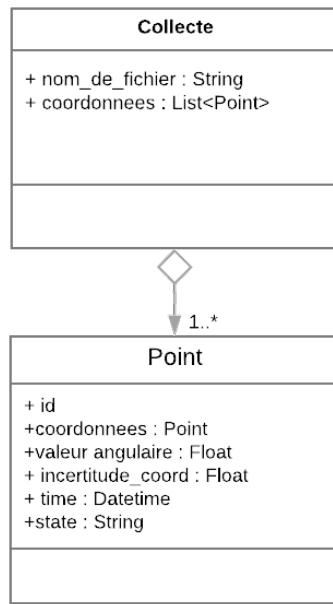
Pour définir un critère de précision, nous avons besoin de calculer le nombre de satellites visibles depuis la position de l'acquisition au temps donné. Pour cela, en milieu urbain notamment, nous avons besoin de l'emprise 3D des bâtiments qui masquent certains signaux. Pour modéliser cette emprise 3D, nous pensons pour le moment simplement extraire les emprises 2D des bâtiments environnants puis de l'élever en 3D avec la donnée de sa hauteur, en le considérant donc comme un parallélépipède rectangle. Cette modélisation est simpliste, mais demeure notre meilleure option sans modèle 3D de ces bâtiments.

F5. Affichage des directions récepteur – satellites à partir de la trajectoire et des éphémérides

Après avoir représenté en 3D les bâtiments, l'objectif est d'afficher la trajectoire des ondes entre les récepteurs et satellites. En réalité, ce sont des fronts d'onde qui sont émis, mais une estimation des trajectoires (tracé d'un vecteur) permet de savoir si ces derniers heurtent un bâtiment. L'estimation de ces trajectoires va se faire à partir des éphémérides radiodiffusées récupérées sur le serveur du RGP et des positions des stations.

F6. Affichage de l'orientation du récepteur via les valeurs angulaires mesurées

Lors des acquisitions, un relevé GNSS et photogrammétrique est effectué. Ces derniers permettent d'associer un trajet 3D avec des images géoréférencées. Pour cela, obtenir l'orientation du récepteur est nécessaire. Cette dernière est obtenue à l'aide d'un accéléromètre.

**FIGURE 2 :** Processus**FIGURE 3 :** Diagramme de classe d'entrée

3.1 Principe de fonctionnement

Ainsi, on prend un fichier CSV en entrée, et on obtient en sortie un fichier KML (figure 2). CSV2KML représente la fonction générale traitant les demandes de fonctionnalités à ajouter au KML.

Le fichier CSV est créé suite à une prise de mesure. Chaque ligne de ce fichier correspond à un point mesuré tandis que les colonnes correspondent aux différents attributs initiaux détaillés dans la figure 3 ci-dessous.

Le fichier KML créé et ses différents composants sont décrits par ce diagramme de classe (figure 4).

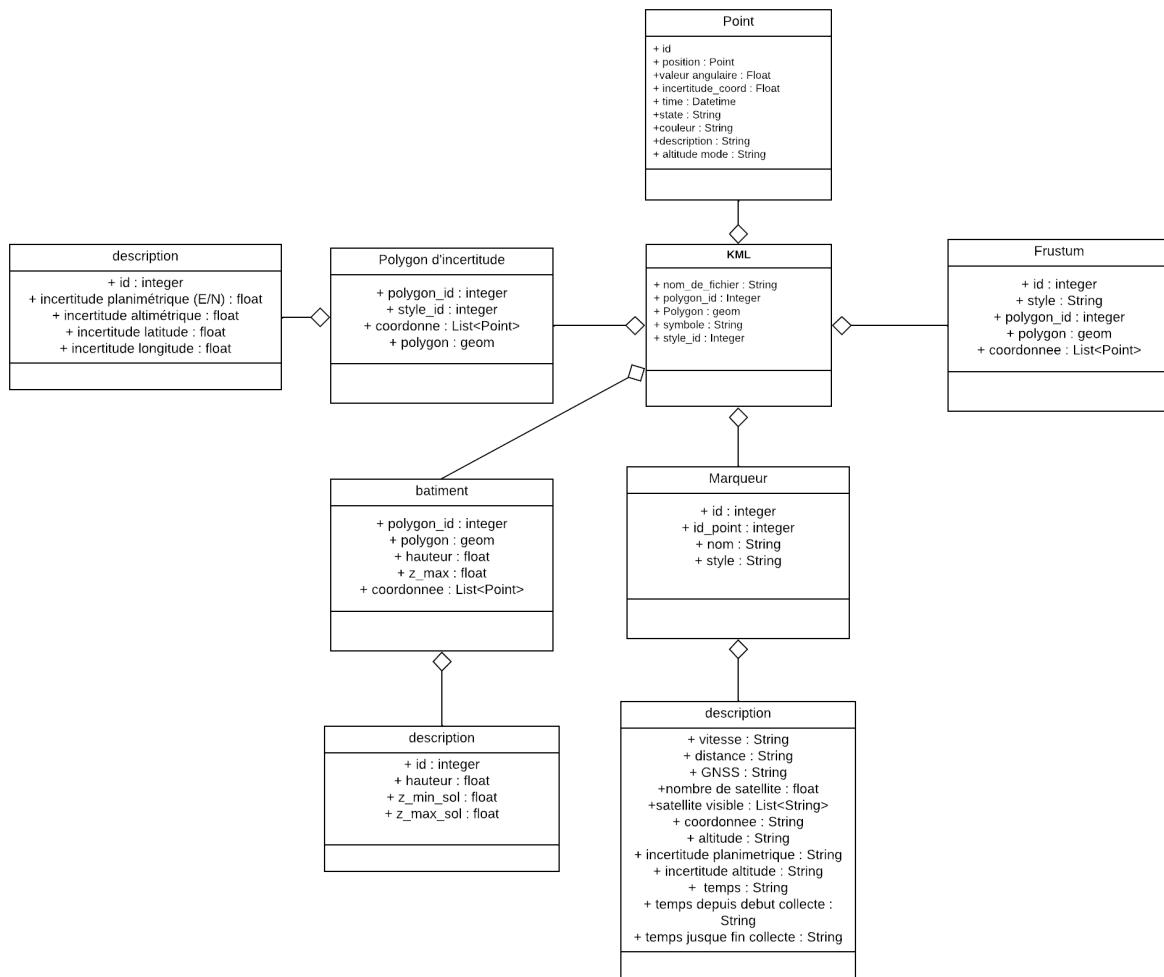


FIGURE 4 : Diagramme de classe de sortie

4 Présentation des fonctionnalités

Le produit développé permet donc d'utiliser indépendamment (ou conjointement) les 6 fonctionnalités évoquées. Un détail du mode d'implémentation est donné dans le [Rapport Technique](#) disponible sur le [git](#).

Chacune des fonctionnalités nécessite 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, cartésien 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, intégrer des objets et 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 F.1 Affichage des attributs associés à chaque position

La fonctionnalité principale du projet permet d'afficher les points mesurés, dont la couleur est paramétrée par le statut de mesure (*RTK float*, *RTK fixed*, *DGNSS*, *None*). Ensuite dans la navigation, l'utilisateur peut sélectionner un point et afficher ses attributs (comme la vitesse, la position, etc) (Figure 5). Cette description est amenée à évoluer en fonction des fonctionnalités utilisées.

L'affichage des points est géré par plusieurs paramètres, qui permettent par exemple de décimer le nombre de points.

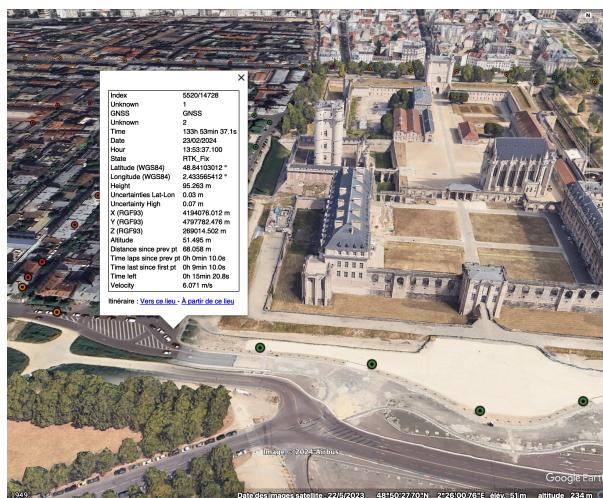


FIGURE 5 : Affichage des points et de leurs attributs - visualisé sur Google Earth

4.2 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. 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 (cohérente avec celle des points (voir 4.1)).

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

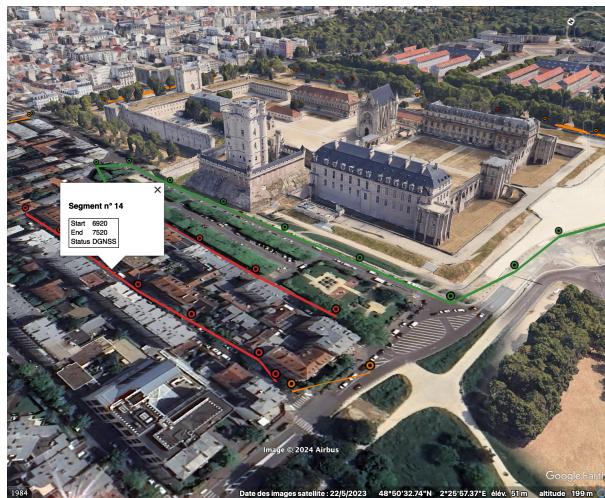


FIGURE 6 : Affichage des segments et de leurs attributs - visualisé sur Google Earth

4.3 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. 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 (et confirmée par le commanditaire) de représenter ces incertitudes par des pyramides à base rectangulaire.

La base de la pyramide est déterminée par l'incertitude planimétrique et la hauteur par l'incertitude altimétrique (Figure 7). Pour rendre la visualisation plus facile, il est possible de définir un seuil d'incertitude (planimétrique ou altimétrique) à partir de laquelle 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.

Pour finir, lors de l'affichage, les intervalles sont colorés avec le même code couleur que le point auxquelles elles sont associées (Figure 8).

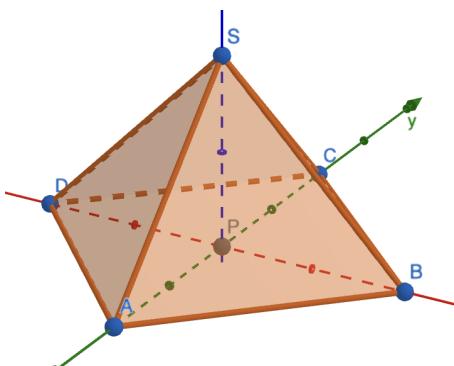


FIGURE 7 : $\|\vec{PB}\|$ l'incertitude Nord-Sud, $\|\vec{PC}\|$ l'incertitude Est-Ouest, et $\|\vec{PS}\| = h$ l'incertitude altimétrique avec P le point mesuré.

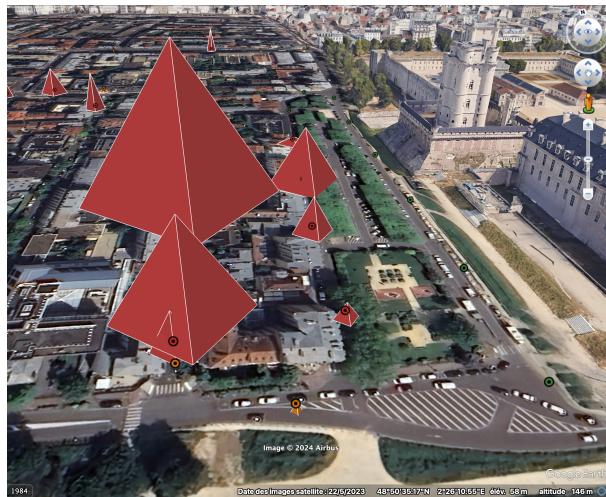


FIGURE 8 : Affichage des intervalles de confiance - visualisé sur Google Earth

4.4 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. 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.

Pour afficher les bâtiments, l'utilisateur doit fournir au programme un fichier *Shapefile* provenant de la BD TOPO v3.0. À partir de cela, le programme sélectionne les bâtiments autour du chantier et les ajoute au KML. L'utilisateur peut ensuite observer les bâtiments, sur le chantier (Figure 9).



FIGURE 9 : Affichage des bâtiments environnants - visualisé sur Google Earth

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

Le premier objectif de cette fonctionnalité permettait de repérer les satellites dont la vision est obstruée par un bâtiment. Cependant, tracer un rayon pour chaque satellite observé aurait rendu la visualisation trop complexe. Nous avons donc modifié le principe pour donner à l'utilisateur une liste des satellites observés sur chaque point (Figure 10).

Pour cela, l'utilisateur fournit les fichiers Rinex d'observation et de navigation enregistrés lors des

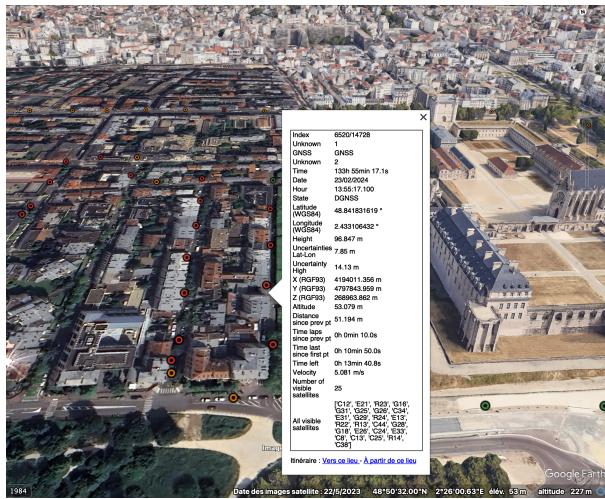


FIGURE 10 : Description des points modifiés pour indiquer les satellites observés - visualisé sur Google Earth

mesures.

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.

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

Lors d'une acquisition en photogrammétrie, on peut coupler à la caméra un capteur GNSS pour positionner les photos. La dernière fonctionnalité développée permet d'utiliser des informations d'orientation du capteur pour représenter les prises de vues. Cette visualisation peut être paramétrée avec la longueur focale, la taille du capteur et la distance de prise de vue.

La visualisation se réalise par le biais d'un frustum, qui représente l'angle de prise de vue depuis le point (Figure 11).

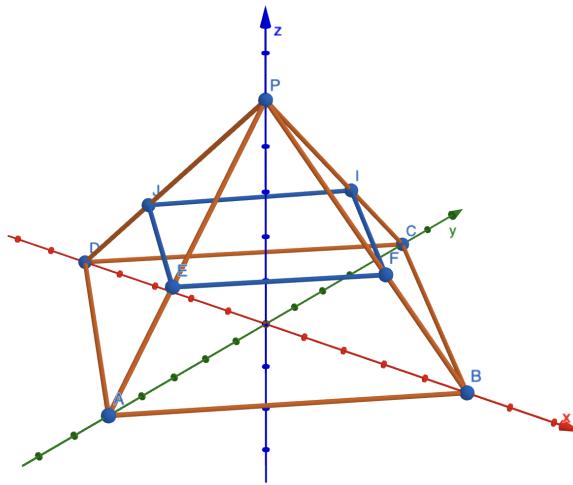


FIGURE 11 : Exemple de Frustum : P le point, $EFIJ$ le capteur, et $ABCD$ l'écran.

Les frustums générés sont donc ensuite intégrés au fichier KML et peuvent être observés par l'utilisateur (Figure 12).

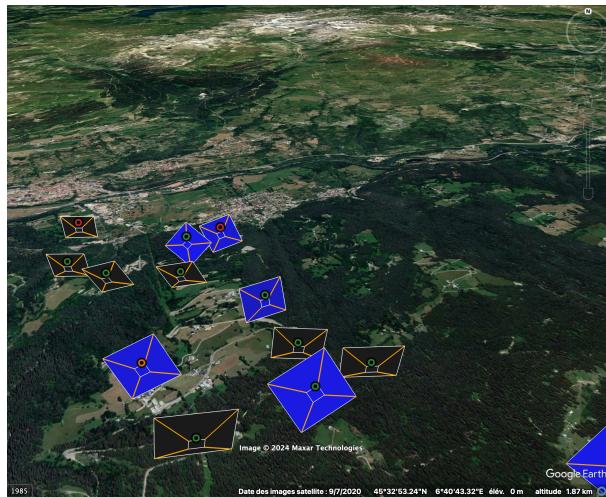


FIGURE 12 : Visualisation des frustums de prises de vues - visualisé sur Google Earth

4.7 Outil global

L'ensemble des fonctionnalités peut être utilisé séparément ou simultanément dans le visualiseur. Les objets créés sont séparés dans des sous-dossiers différents pour permettre une modulation interactive de l'affichage. De plus, chaque type d'objet ajouté peut être paramétré et ajuster pour s'accorder aux besoins de l'utilisateur.

Ainsi, dans sa version finale, notre projet permet à l'utilisateur d'observer certains indicateurs et de les mettre en corrélation visuelle pour interpréter la qualité de ses mesures.

5 Réalisation et suivi du projet

5.1 Risques

Afin de mener à bien notre projet dans le temps, nous avons cherché à identifier les risques potentiels. Nous avons estimé ces risques :

- des problèmes de compatibilité entre les différentes librairies python ;
- une mauvaise compréhension et utilisation des données,
- une quantité de donnée trop importante à gérer,
- des temps de calcul trop long,
- des problèmes de gestion du temps,
- une mésentente entre les différents membres du projet.

La matrice des risques (Figure 15) met en valeur ces potentiels risques. Celle-ci a évolué au cours du projet.

Finalement, lors de l'avancement du projet, beaucoup de risques que nous avions envisagés ne nous ont pas posé de problème. Les risques majeurs que nous avons eu à gérer concernent les problèmes de lourdeurs dans les données comme l'affichage du nombre de mesures ou encore certains temps de calculs. On peut citer comme exemple de résolution celui de la quantité de données. En effet, des acquisitions de

mesure toutes les 10ms fournissaient un nombre de mesure très important et donc difficilement lisible dans le visualiseur. C'est pourquoi nous avons choisi de faire une décimation de données qui est modifiable par l'utilisateur.

5.2 Planning suivi versus planning prévisionnel

Découpé en 6 fonctionnalités, nous avions initialement pensé à traiter les fonctionnalités par groupe de deux. Donc deux fonctionnalités sont traitée en même temps par deux groupe. Pour chaque fonctionnalité, nous voulons une phase de codage (trois semaines), une phase de test et une phase de fusion avec le reste du code (une semaine chacune). Il faut ensuite ajouter les phases finales de tests et les phases de rédaction des différents livrables (rapport d'analyse finale et rapport technique, documentation du projet, readme...).

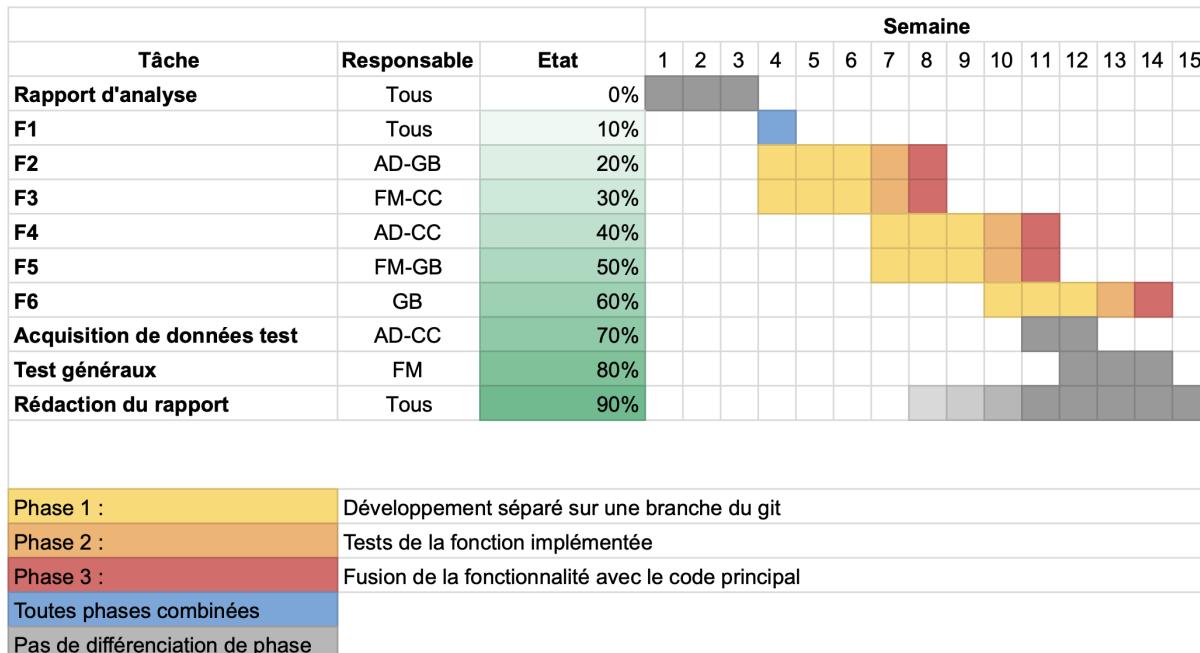


FIGURE 13 : Planning prévisionnel

Le planning final ressemble au planning que nous avions anticipé, à ceci près que la difficulté croissante des fonctionnalités nous a obligé à passer moins de temps sur les premières et plus sur les dernières. Certaines phases de tests ont aussi révélé des failles dans le code qui ont demandé une semaine supplémentaire d'implémentation. Ce diagramme ne permet pas de l'illustrer, mais les groupes de deux se sont généralement scindés avant la fin de l'implémentation d'une fonctionnalité, soit pour entamer la prochaine soit pour apporter un point de vue nouveau dans l'autre groupe lors de la phase de test ou alors pour rédiger les parties relatives à la fonctionnalité en question sur les différents livrables.

Tâche	Responsable	Etat	Semaine													
			1	2	3	4	5	6	7	8	9	10	11	12	13	14
Rapport d'analyse	Tous	100%														
F1	Tous	100%														
F2	AD-GB	100%														
F3	FM-CC	100%														
F4	AD-CC	100%														
F5	FM-GB	100%														
F6	GB	100%														
Acquisition de données test	AD-CC	0%														
Test généraux	Tous	100%														
Rédaction du rapport	Tous	100%														

Phase 1 :	Développement séparé sur une branche du git
Phase 2 :	Tests de la fonction implémentée
Phase 3 :	Fusion de la fonctionnalité finalisée avec le code principal
Toutes phases combinées	
Pas de différenciation de phase	

FIGURE 14 : Planning final

Nature du risque	Probabilité	Conséquence	Solution prévisionnelle/ corrective	Impact	Evolution
Incompatibilité entre certaines librairies	Faible	Retard dans le développement	Analyser les dépendances	significatif	
Défauts dans les données sources (manque de données, acquisitions ratées, ...)	Modérée	Mauvaise interprétation des résultats/ retard dans le développement	Discuter de la fiabilité des données avec le commanditaire	significatif	
Difficulté d'installation des librairies	Forte	Retard dans le développement	Consulter la documentation. Demander conseil au commanditaire/ professeurs	peu significatif	
Modélisation des bâtiments en 3D trop simpliste	Modérée	Mauvaise visualisation des modèles	Effectuer une modélisation qualitative	peu significatif	
Irrespect de l'emploi du temps déjà élaboré	Modérée	Retard dans le développement	Valoriser une bonne gestion du temps et des tâches	significatif	
Passer trop de temps sur une tâche en étant bloqué	Forte	Retard dans le développement/ frustration dans le groupe	Entraide afin de traiter les difficultés	fort	

FIGURE 15 : Matrice des risques

6 Conclusion

Ainsi, une analyse approfondie des besoins, des contraintes et des objectifs nous a permis d'identifier les éléments essentiels à la conception et à la mise en œuvre réussie de ce projet. Même si nous n'avons pas pu anticiper toutes les difficultés auxquelles nous avons dû faire face, nous avons réussi à identifier les principales. Cela nous a permis de répondre à toutes les exigences de notre commanditaire dans le temps imparti.

Notre vision du projet a évolué au fur et à mesure de l'avancement, car nous en apprenions plus à chaque fois sur ce que permet le format de fichier KML, et sur ce à quoi le rendu final devait ressembler. Cela a eu pour effet d'écartier définitivement certains risques et de réévaluer la probabilité de rencontrer certains autres.

Nous considérons ce projet comme une réussite, et ce pour plusieurs raisons. Premièrement parce que l'ensemble des exigences de notre commanditaire ont été satisfaites. Mais aussi, car chaque membre du groupe a joué son rôle et a permis de tenir les délais demandés. En outre, ce projet nous a apporté de nombreuses choses, que ce soit sur l'aspect technique du développement informatique ou des mesures GNSS, mais aussi sur la manière dont un projet se déroule et s'organise.

7 Références et outils

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/>