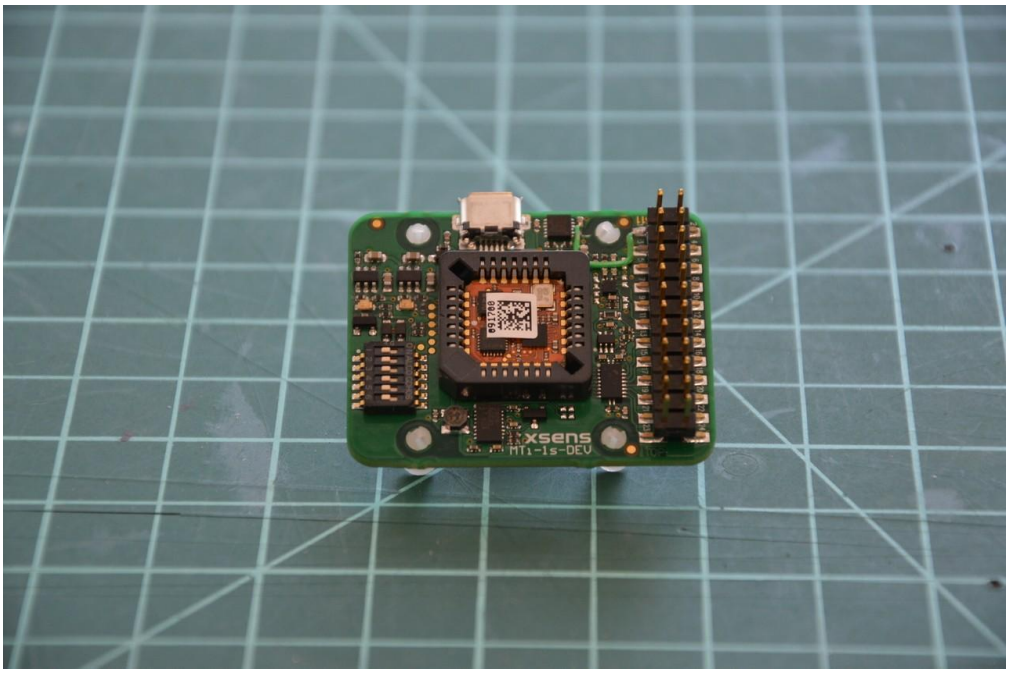
IUT Angers-Cholet 4, boulevard Lavoisier - BP 42018 49016 - ANGERS Cedex

Projet Web IMU 3D



Projet réalisé par

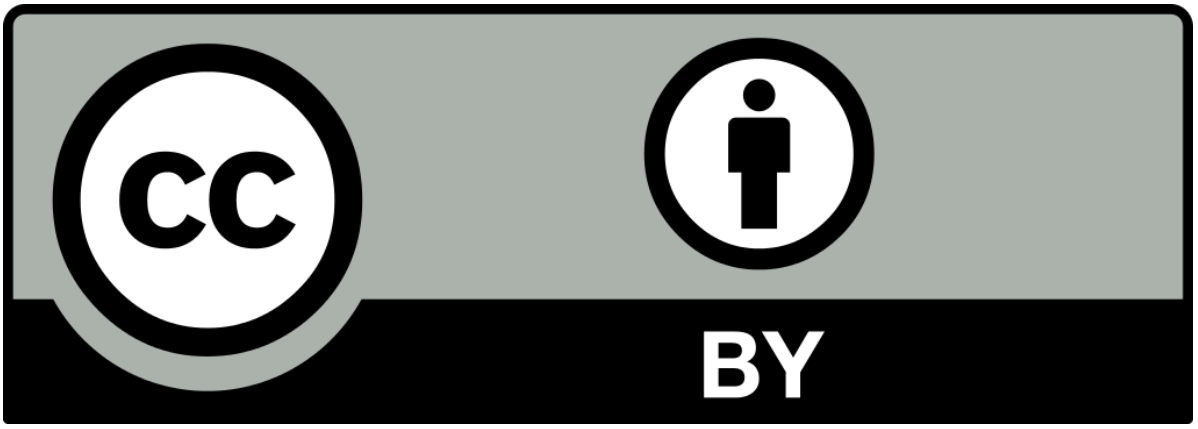
Enzo FORESTIER Maël RIGAUDEAU

Projet encadré par Philippe LUCIDARME

DÉCHARGE DE RESPONSABILITÉS

Ce rapport présente le travail réalisé par un groupe d'étudiants dans le cadre d'un projet pédagogique. Les auteurs et l'Université d'Angers ne garantissent pas que l'information, les documents, la méthodologie et le matériel présentés dans ce document soient complets, conformes à l'état de l'art et exacts ni n'assurent en toutes circonstances la sécurité des biens, des personnes et des utilisateurs. Les auteurs et l'Université d'Angers ne seront pas tenus responsables des dommages éventuels qui pourraient résulter de l'utilisation du contenu du présent rapport

LICENCE



REMERCIEMENTS

Tout d’abord, nous tenons à remercier tout particulièrement et à témoigner toute notre reconnaissance aux personnes suivantes, pour leur dévouement et leur soutien dans la concrétisation de ce projet :

M. Philippe LUCIDARME, responsable projet, pour ses conseils éclairés, sa patience, sa disponibilité et pour la confiance qu’il nous a accordée dès l’ébauche du projet et tout au long de ces deux mois.

M. Lionel LEDUC, responsable projet, pour nous avoir accordé toute la confiance nécessaire pour élaborer ce projet librement.

L’IUT et l’ensemble des enseignants pour leur coopération professionnelle tout au long de cette expérience et pour avoir partagé avec nous, une partie de leurs savoir-faire et de leurs expériences professionnelles

Table des matières

I. INTRODUCTION

Dans le cadre de notre seconde année en BUT Génie Electrique et Informatique Industrielle (GEII) à l'IUT d'Angers, il nous est proposé un projet de 3 mois nous permettant de mettre en pratique nos connaissances et nos compétences professionnelles au travers d'un cahier des charges ayant pour finalité la conception et le développement d'une application industrielle en accord avec nos intérêts professionnels.

Ayant une passion commune pour la programmation et le développement, notre groupe composé de Maël RIGAUDEAU et Enzo FORESTIER, a saisi l'opportunité d'exploiter cet intérêt commun pour réaliser un projet innovant au responsable de l'année M. Philippe LUCIDARME.

Nous avons alors réalisé un projet informatique visant à modéliser une centrale inertielle en 3D sur une page web. L'objectif principal de ce projet était de créer une interface conviviale et interactive permettant de visualiser en temps réel les données capturées par la centrale inertielle. Pour atteindre cet objectif, nous avons utilisé Visual Studio Code, un environnement de développement intégré puissant et polyvalent, ainsi que la bibliothèque Flask, un Framework Web léger et flexible pour Python.

II. Objectifs du projet

1.Contexte

Lors du premier semestre nous avons eu l'opportunité de travailler sur un capteur inertiel (capteur qui permet de situer un objet dans l'espace) avec pour objectif de créer une librairie en C++ nous permettant de récupérer les angles d'Euler en temps réel. Après avoir réussi notre projet nous avons réfléchi à la suite de ce premier projet. En effet celui-ci nous avait beaucoup plus et nous voulons donc le continuer au semestre 4.

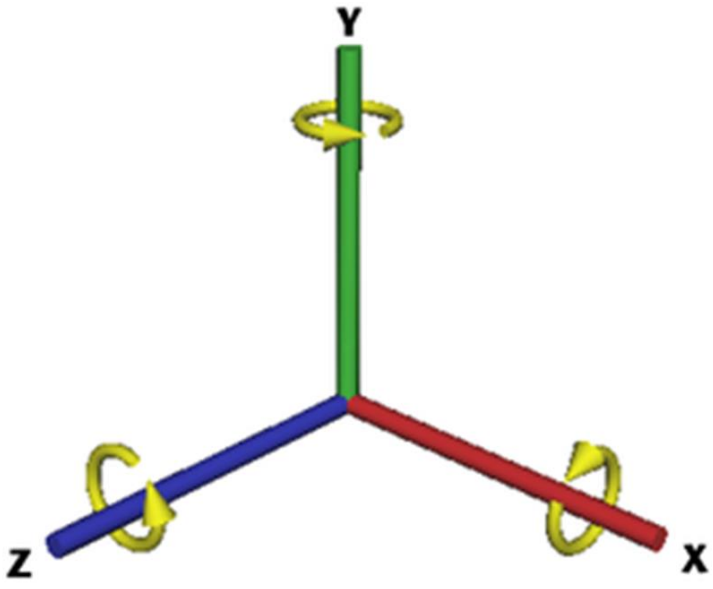
2. Cahier des charges

Notre cahier des charges est le suivant : Créer un site web qui affiche la modélisation 3D de la carte X-SENS MTI-3 en fonction des angles d'Euler.

Mais que sont les angles d'Euler ?

Les angles d'Euler sont des angles pour décrire la position d'un solide dans l'espace. Il y en a 3 :

- le tangage qui correspond à une rotation autour de l'axe X,
- le lacet qui lui correspond à une rotation autour de Y,
- le roulis qui lui correspond à une rotation autour de Z.



Euler rotation interpolation

Pour mettre à bien ce projet nous allons donc utiliser :

-La centrale inertielle MTI-3

-Le logiciel Visual studio code

A noter que nous effectuons ce projet sous Windows.

III. Schéma Général

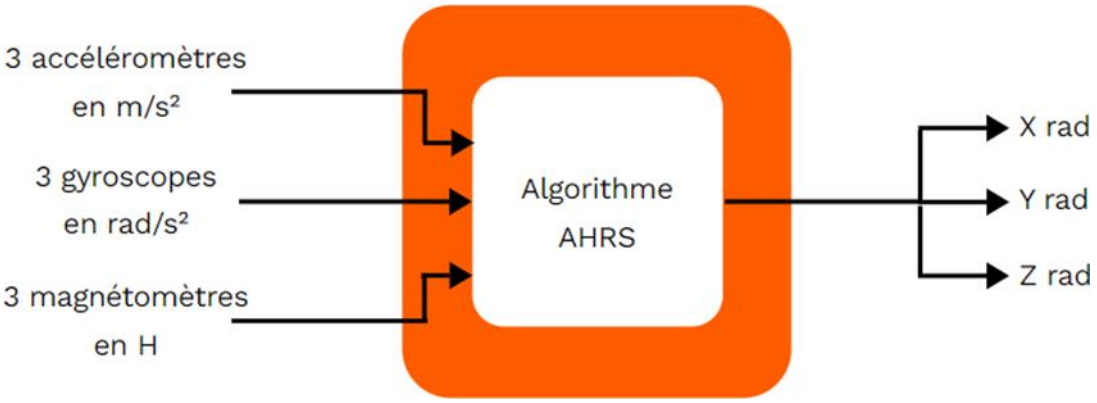
1.Fonctionnement de la centrale inertielle mti-3

La centrale inertielle MTi-3 fabriquée par la société Xsens est une IMU 9 axes qui embarque les capteurs suivants :

Accéléromètre 3 axes (mesure l'accélération linéaire le long des axes X, Y et Z) ;
Gyroscope 3 axes (mesure l'accélération angulaire autour des axes X, Y et Z) ;
Magnétomètre 3 axes (mesure le champ magnétique le long des axes X, Y et Z) ;
Le plus intéressant avec cette centrale inertielle est l'algorithme de fusion de données. L'algorithme AHRS fusionne les données des 9 capteurs pour fournir directement l'orientation du capteur (roulis / tangage / lacet).

L'acronyme AHRS signifie Attitude and Heading Reference System.
Il s'agit d'un algorithme ou d'un système, généralement composé de capteurs 3 axes, permettant d'estimer l'orientation d'un mobile (typiquement un avion ou un drone) dans l'espace. L'orientation est composée de roulis, tangage et lacet.

Voici un schéma de fonctionnement de la centrale inertielle mti-3 :



2.Schéma générale de la modélisation 3D

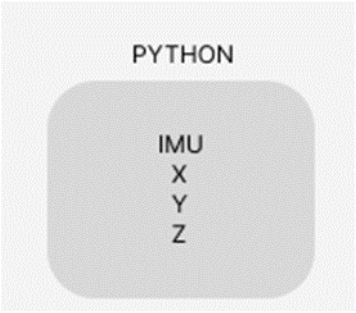
Dans le cadre de notre projet actuel, nous sommes confrontés à la tâche de concevoir et développer un système basé sur les données provenant de la centrale inertielle de la carte Xsens. Ce projet constitue une extension directe du travail que nous avons réalisé lors du semestre 3, où nous avons également exploité les informations fournies par cette centrale inertielle.

Cependant, dans cette nouvelle itération du projet, nous avons pris la décision d'adopter de nouvelles approches technologiques. Tout d'abord, nous avons opté pour l'utilisation d'un logiciel de programmation différent, à savoir Visual Studio Code. Ce choix découle de notre désir d'explorer et d'exploiter les fonctionnalités avancées offertes par cet environnement de développement intégré (IDE).

En outre, nous avons également décidé d'adopter de nouveaux langages de programmation pour mettre en œuvre notre système. Parmi ces langages, nous avons choisi d'utiliser Python, JavaScript, HTML et CSS. Cette décision repose sur la volonté d'exploiter les avantages spécifiques offerts par chacun de ces langages, et de tirer parti de leurs capacités de manière synergique afin de répondre aux besoins complexes du projet.

Une autre étape importante que nous avons dû franchir pour mettre en place notre environnement de développement était l'installation d'un pilote spécifique sur nos ordinateurs. Ce pilote permet la communication et la lecture des données envoyées par la carte Xsens. Cette étape, bien que technique, s'est avérée essentielle pour établir une connexion solide entre notre système et la centrale inertielle, garantissant ainsi un transfert fluide des données nécessaires à notre projet.

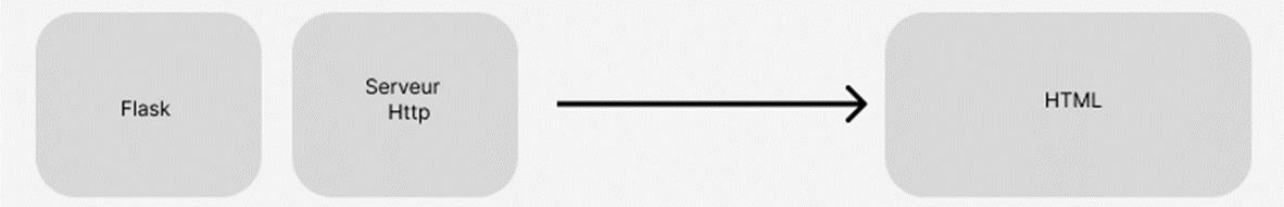
Premièrement, Dans le but de simplifier la communication ultérieure avec le serveur Web, nous avons pris la décision de réécrire notre programme de lecture des trames de la carte Xsens, qui était initialement écrit en C++, en utilisant le langage Python.



Afin de transmettre les données reçues depuis la carte Xsens, nous avons identifié la nécessité de mettre en place un serveur HTTP dédié. Après une évaluation approfondie des options disponibles, nous avons choisi d'utiliser la bibliothèque Flask pour créer notre serveur.

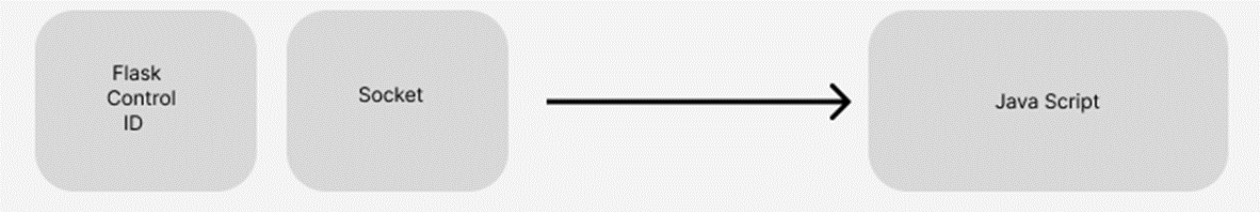
Plusieurs raisons ont motivé notre choix de Flask. Tout d'abord, Flask est réputé pour être facile à utiliser et à mettre en œuvre. Sa syntaxe claire et concise permet de développer rapidement des applications Web, même pour ceux qui n'ont pas une expérience approfondie en développement web. Cette simplicité d'utilisation était un facteur crucial pour nous, car nous souhaitions nous concentrer davantage sur la logique métier et le traitement des données, plutôt que sur la complexité de la mise en place du serveur.

Une fois le serveur HTTP créé, nous avons pu créer une page HTML qui offre une interface conviviale et intuitive pour interagir avec la modélisation 3D. En intégrant les technologies HTML et CSS, nous avons pu créer une mise en page esthétiquement agréable et bien structurée, offrant une expérience utilisateur optimale.

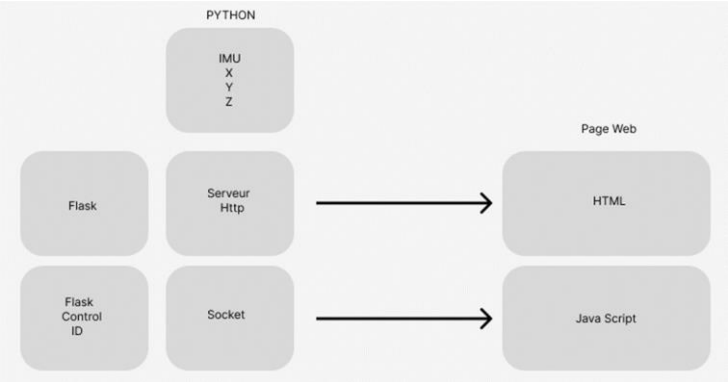


Pour permettre une communication en temps réel entre le serveur et la page HTML, nous avons utilisé la bibliothèque Flask-Socket IO pour mettre en place un canal de transmission de données basé sur les Web Sockets. Cette approche nous a permis de transmettre efficacement les données de la centrale inertielle depuis le serveur vers la page web, sans nécessiter de rafraîchissement de la page.

En utilisant JavaScript, nous avons pu recevoir les données de la centrale inertielle transmises par le serveur et les traiter en temps réel. Ces données ont été utilisées pour mettre à jour la modélisation 3D présente sur la page web. Nous avons utilisé les informations de position, d'orientation et d'autres paramètres pour manipuler les objets 3D dans l'environnement WebGL créé avec Three.js. Cette manipulation en temps réel des objets 3D a permis de visualiser les mouvements et les orientations de la carte Xsens de manière interactive sur la page web.



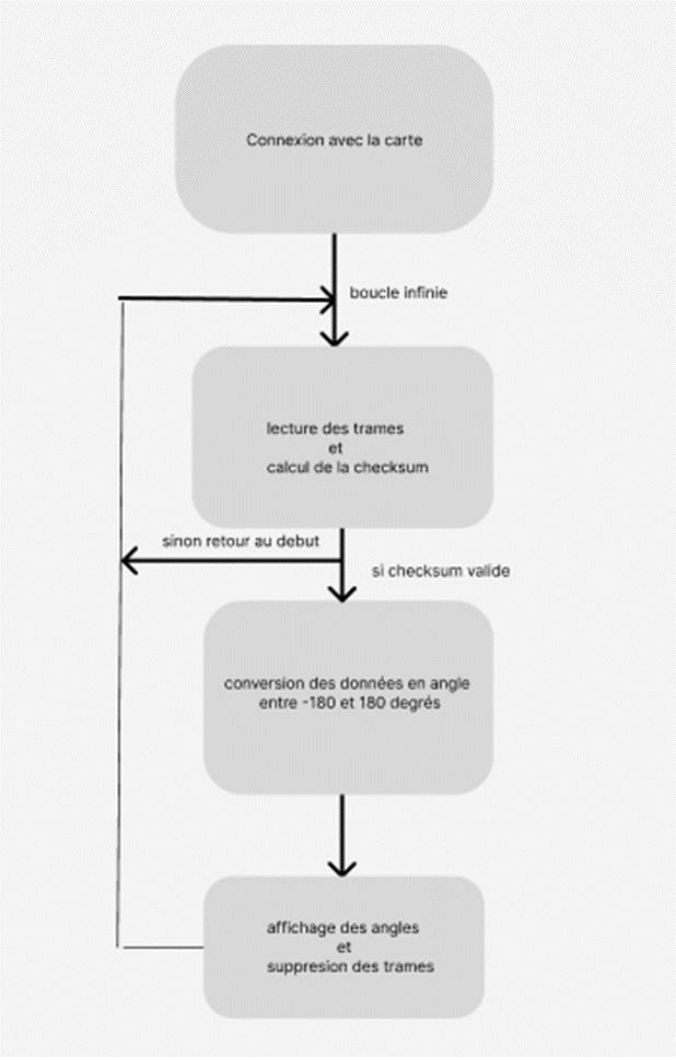
Voici le schéma général de la modélisation 3D complet :


PYTHON
IMU
X
Y
Z
Page Web
Flask
Serveur
Http
HTML
Flask
Control
ID
Socket
Java Script

IV. Livrables

1.Conversion de la réception des données en Python

Pour cette première partie nous avons eu besoin de convertir notre code fait en C++ lors du semestre 3 en python. Pour cela nous avons pris une version du code ou nous n'avions pas créer de classe ni de librairie car cela aurait été trop compliqué à convertir avec le peu de temp que nous avions.
Voici donc le schéma fonctionnel de notre programme :



La première partie du code va consister à se connecter au capteur inertiel. Pour le code en C++ nous utilisions la librairie "serialib" qui nous permettait de se connecter à un port série pour lire les données qu'on recevait. Il était donc nécessaire de trouver une bibliothèque similaire en python. Nous avons donc utilisé la bibliothèque "serial " qui permet de faire la même chose.

Un autre des problèmes que nous avons eu est que notre projet fait en C++ était sous linux. Sous linux la carte se lit si on spécifié le chemin du port mais pas sous Windows ou du moins pas si un driver pour lire la carte n'a pas été installé. Nous avons donc cherché sur le site du constructeur et avons trouvé ce fameux driver.

La suite de la conversion se fit plutôt bien. Une fois la syntaxe de python comprise nous n'avons pas eu d'autre difficulté pour la conversion.

2)Création d'un serveur Flask

Pour la réalisation de notre projet nous avons choisi d'utiliser le Framework Flask pour la création du serveur. Plusieurs raisons ont motivé ce choix.

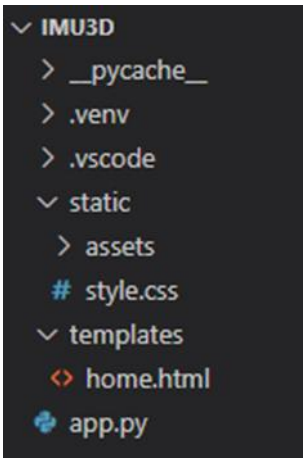
Tout d'abord, Flask est un Framework Web léger et flexible développé en langage Python. Sa simplicité et sa facilité d'utilisation en font un choix idéal pour les projets de petite à moyenne envergure tels que le nôtre. En utilisant Flask, nous pouvions nous concentrer sur l'essentiel de notre application, c'est-à-dire la visualisation des données de la centrale inertielle, sans être alourdis par des fonctionnalités excessives ou complexes.

De plus, Flask offre une grande modularité et permet une grande liberté dans la conception de l'architecture de notre application. Il utilise le principe de routes pour associer des fonctions à des URL spécifiques, ce qui facilite la gestion des différentes pages et des interactions avec l'utilisateur. Cette approche permet une organisation claire et logique du code, favorisant la maintenance et la flexibilité de l'application.

Pour installer Flask, nous avons utilisé l'outil de gestion de paquets de Python, pip. En exécutant simplement la commande "pip Install flask" dans notre environnement de développement, nous avons pu télécharger et installer Flask ainsi que ses dépendances nécessaires. Cette installation a été rapide et sans complications, ce qui témoigne de la facilité d'utilisation et de la popularité de Flask dans la communauté de développement Python.

Notre architecture de projet reposait sur trois composants principaux : les pages HTML, les fichiers Python et les styles CSS. Les pages HTML étaient utilisées pour définir la structure et le contenu de notre application web. Les fichiers Python, quant à eux, étaient utilisés pour gérer les routes, les interactions avec la centrale inertielle et les opérations de traitement des données. Les styles CSS étaient utilisés pour ajouter des éléments de design et de mise en page à nos pages HTML.

Voici comment se présente notre architecture de projet :



Une fois Flask installé, nous avons pu commencer à construire notre serveur en utilisant ses fonctionnalités. Flask suit le modèle de programmation de l'architecture web appelé MVC (Modèle-Vue-Contrôleur), où les routes définissent les vues qui seront affichées à l'utilisateur et les contrôleurs qui gèrent les interactions avec les données. Cette approche modulaire et intuitive nous a permis de créer rapidement des routes pour notre application, en associant chaque URL à une fonction spécifique qui traite les requêtes et retourne les réponses correspondantes.

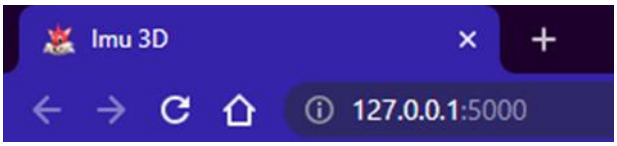
Voici un exemple de route que nous avons pu utiliser :

```python
# Route principale pour la page d'accueil
@app.route("/")
def home():
    return render_template("home.html")
```

Pour lancer le serveur Flask, nous avons utilisé la commande suivante dans le terminal de Visual Studio Code : python -m flask --debug run. Cependant, il est important de noter qu'il est impératif d'être dans l'environnement virtuel précédemment créé pour exécuter cette commande avec succès.

Pour activer l'environnement virtuel, nous avons utilisé la commande suivante dans le terminal de Visual Studio Code : c:/Users/Moi/Desktop/Imu3D/.venv/Scripts/Activate.ps1. Cette commande active l'environnement virtuel spécifique à notre projet, qui contient toutes les dépendances nécessaires pour exécuter le serveur Flask.

Une fois le serveur lancé avec succès, nous avons pu accéder à notre application web en ouvrant un navigateur et en entrant l'URL appropriée, ici 127.0.0.1:5000.



3)Communication entre les données et le serveur

Pour communiquer entre les données et le serveur nous avons choisi d'envoyer les données par protocole Web sockets, et de les recevoir avec Javascript comme en témoigne notre schéma général :

3.1) Flask-Socket IO

Nous avons choisi pour notre projet d'utiliser des Web Sockets, elles sont un protocole de communication bidirectionnelle en temps réel, utilisé pour établir une connexion persistante entre un serveur et un client.
Pour assurer la communication en temps réel entre les données provenant de la centrale inertielle et le serveur, nous avons utilisé la bibliothèque Flask-Socket IO. Cette bibliothèque facilite la communication bidirectionnelle entre le serveur et le client, permettant ainsi de transmettre les données capturées par la centrale inertielle en temps réel.

Dans notre code Python, nous avons mis en place des gestionnaires d'événements pour gérer la connexion et la déconnexion des clients socket. Lorsqu'un client socket se connecte au serveur, le gestionnaire d'événements "@socketio. On("connect")" est déclenché et exécute les actions définies à l'intérieur. Par exemple, dans notre cas, nous avons simplement affiché un message indiquant que le client s'est connecté en utilisant la fonction "print".

De même, lorsque le client socket se déconnecte du serveur, le gestionnaire d'événements "@socketio. On("disconnect")" est déclenché et exécute les actions correspondantes. Dans notre exemple, nous avons affiché un message indiquant que le client s'est déconnecté.

Une fois que nous avons reçu les données de la centrale inertielle et que nous les avons traitées dans notre code Python, nous avons utilisé la ligne suivante pour les envoyer au client socket :

```
socketio.emit("sensor_data", {"x": a1, "y": b1, "z": c1})
```

Cette ligne de code envoie un événement nommé "sensor_data" au client socket avec les données capturées de la centrale inertielle. Les données sont envoyées sous la forme d'un dictionnaire avec des clés correspondant aux axes de l'accélération ou de la rotation, et des valeurs représentant les mesures spécifiques.

L'utilisation de Flask-Socket IO nous a permis d'établir une communication en temps réel entre le serveur et le client, permettant ainsi d'afficher les données de la centrale inertielle presque instantanément sur la page web. Cela a amélioré l'expérience utilisateur en fournissant des mises à jour en temps réel et en offrant une interaction fluide avec les données capturées.

3.2) JavaScript

La réception des données à partir du serveur Flask via Web Sockets est effectuée à l'aide d'un script JavaScript côté client. Voici comment ce script fonctionne :

Tout d'abord, une connexion est établie avec le serveur Socket.IO en utilisant la fonction io. Connect. L'adresse du serveur est déterminée en utilisant document. Domain et location. Port pour garantir la connexion au bon emplacement. Par exemple :

```
var socket = io.connect(
  "http://" + document.domain + ":" + location.port
); // Connexion au serveur Socket.IO
```

Ensuite, un gestionnaire d'événements est défini pour le moment où la connexion avec le serveur est établie. Dans cet exemple, l'événement "connect" est utilisé. Lorsque la connexion est établie, le gestionnaire d'événements est déclenché et exécute les actions spécifiées à l'intérieur. Dans ce cas, une émission d'événement "read_serial" est envoyée au serveur pour signaler la connexion réussie. Par exemple :

```
socket.on("connect", function () {
  socket.emit("read_serial", { data: "I'm connected!" }); // Émission d'un événement "read_serial" au serveur
  // console.log("connecté");
});
```

Enfin, un autre gestionnaire d'événements est défini pour l'événement "sensor_data". Ce gestionnaire est appelé lorsque le serveur envoie des données à travers les Web Sockets. Les données sont reçues en tant que msg et sont ensuite utilisées pour mettre à jour le contenu des éléments HTML spécifiés. Dans cet exemple, les données reçues sont des valeurs d'axes (x, y, z) provenant de la centrale inertielle. Ces valeurs sont utilisées pour mettre à jour les éléments HTML correspondants en utilisant textContent et toFixed pour formater les données. Par exemple :

```
socket.on("sensor_data", function (msg) {
  // console.log("socket reçu");
  document.getElementById("x").textContent = msg.x.toFixed(4); // Mise à jour du contenu de l'élément HTML avec l'ID "x"
  document.getElementById("y").textContent = msg.y.toFixed(4); // Mise à jour du contenu de l'élément HTML avec l'ID "y"
  document.getElementById("z").textContent = msg.z.toFixed(4); // Mise à jour du contenu de l'élément HTML avec l'ID "z"
});
```

Ainsi, lorsque le serveur envoie des données à travers les Web Sockets, le gestionnaire d'événements "sensor_data" est déclenché et met à jour les éléments HTML correspondants avec les valeurs reçues.

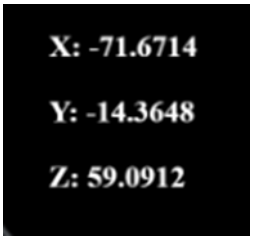
4)Création du site Web et modélisation 3D de la carte

4.1) Site Web

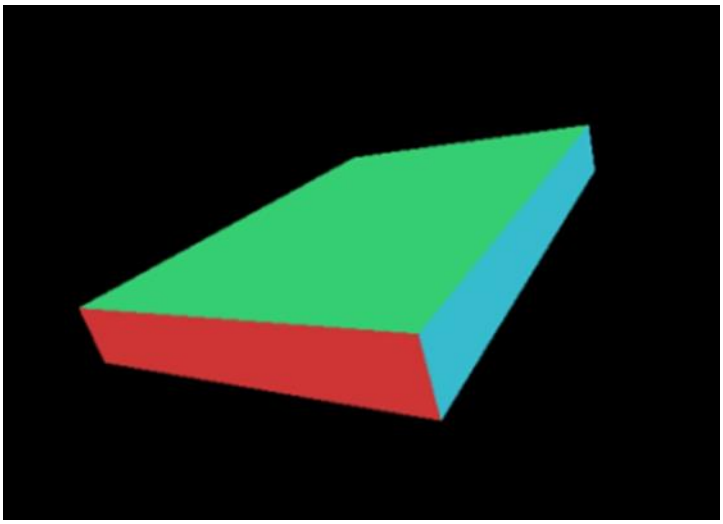
Le site web a été conçu en utilisant les technologies HTML, CSS et JavaScript. Le code HTML fournit une structure de base pour la page web, tandis que le code CSS est utilisé pour la mise en forme et l'apparence visuelle. Le code JavaScript est responsable de l'interactivité et de la modélisation en 3D.

Lorsque l'utilisateur accède au site web, il est accueilli par un titre attrayant « Modélisation 3D d'une centrale inertielle”.

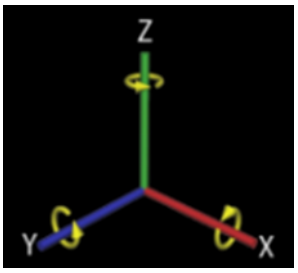
L'interface web présente également une section de données affichant les valeurs des coordonnées de la centrale inertielle dans les axes X, Y et Z. Ces valeurs sont mises à jour en temps réel à l'aide de la technologie Socket.IO, qui permet une communication bidirectionnelle entre le serveur et le client.



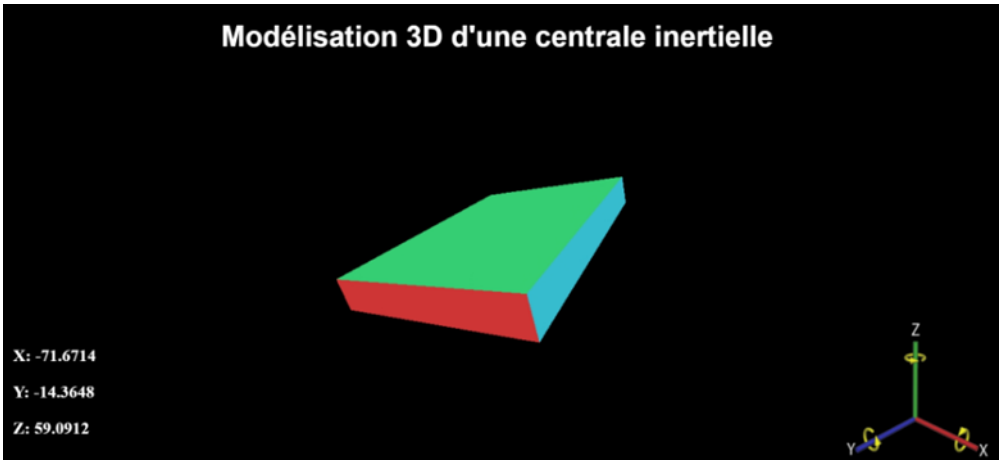
Une autre section du site web contient une représentation visuelle en 3D de la centrale inertielle. Cette représentation est réalisée à l'aide de la bibliothèque Three.js, qui facilite la création et la manipulation d'objets en 3D dans un environnement WebGL. Le modèle de la centrale inertielle est représenté par un cube tridimensionnel.



Pour finir, une image a été placé en bas à droite de la page pour indiquer comment regarder la modélisation 3D.



Voici finalement comment se présente notre page web :



4.2) Modélisation en 3D de la centrale inertielle

La modélisation en 3D de la centrale inertielle est réalisée à l'aide de la bibliothèque Three.js. Lorsque les données provenant de la centrale inertielle sont reçues via la connexion Socket.IO, les angles de rotation correspondants sont extraits et convertis en radians.

```
socket.on("sensor_data", function (msg) {
  // Convertir les angles en radians
  var angleX = THREE.MathUtils.degToRad(msg.x);
  var angleY = THREE.MathUtils.degToRad(msg.y);
  var angleZ = THREE.MathUtils.degToRad(msg.z);
```

Le cube tridimensionnel représentant la centrale inertielle est créé à l'aide de la classe THREE.BoxGeometry, avec des dimensions spécifiées. Chaque face du cube est associée à un matériau différent pour permettre une distinction visuelle des axes de rotation. Les couleurs choisies sont le rouge pour l'axe X, le bleu pour l'axe Y et le vert pour l'axe Z.

```
var geometry = new THREE.BoxGeometry(4, 3, 0.5);

// Définir une texture différente pour chaque face du cube
var materials = [
  new THREE.MeshBasicMaterial({ color: 0xce3636 }), // Face X - rouge
  new THREE.MeshBasicMaterial({ color: 0xce3636 }), // Face X - rouge
  new THREE.MeshBasicMaterial({ color: 0x36bcce }), // Face Y - bleu
  new THREE.MeshBasicMaterial({ color: 0x36bcce }), // Face Y - bleu
  new THREE.MeshBasicMaterial({ color: 0x36ce73 }), // Face Z - vert
  new THREE.MeshBasicMaterial({ color: 0x36ce73 }), // Face Z - vert
];
```

Les valeurs des angles de rotation reçues sont ensuite appliquées aux propriétés de rotation du cube à l'aide des propriétés rotation, rotation. Y et rotation. Cela permet de faire tourner le cube dans l'espace 3D en fonction des mouvements de la centrale inertielle.

```
  // Appliquer les angles au cube
  cube.rotation.x = angleX;
  cube.rotation.y = angleY;
  cube.rotation.z = angleZ;
});
```

Enfin, la scène 3D est rendue à l'aide du moteur de rendu WebGL fourni par Three.js, et le résultat est affiché sur la page web.

L'animation de la modélisation en 3D est gérée par la fonction animate (), qui utilise la fonction requestAnimationFrame pour mettre à jour en continu l'affichage de la scène 3D.

IV. BILAN DU PROJET

Ce projet nous a beaucoup apporté. Il nous a permis de comprendre le type de mission que l'on pourrait avoir dans le monde professionnel. Nous avons donc pu améliorer nos compétences techniques en apprenant à coder en Python, java script, HTML et CSS. Nous avons aussi amélioré des compétences plus globales telles que l'autonomie, la responsabilité, la ponctualité et l'entraide. Compétences qui sont indispensables dans le monde professionnel. Le format proposé avec l'aide des professeurs quand on le souhaite sans pour autant être trop guidé est parfait. Le fait de devoir présenter nos avancés en faisant des oraux ou des rapports nous met en condition de vrai projet dans le monde professionnel. Nous avons aussi pu approfondir trois compétences acquises au sein de l'iut que sont : Concevoir, vérifier et maintenir.

D'un point de vue plus personnel, travailler sur ce projet m'a aidé à y voir plus clair dans le domaine dans lequel je veux évoluer. En effet la découverte de nouveaux langages de programmation et le faite de devoir régler toute sorte de problème avec des résultats observables me plait et je pense m'orienter vers l'informatique dans la suite de mes études. (Enzo)

Personnellement, ce projet a été une expérience enrichissante qui m'a permis d'explorer de nouvelles technologies et de consolider mes connaissances en programmation. J'ai pu mettre en pratique mes compétences en utilisant des choix technologiques innovants tels que Visual Studio Code, Python, JavaScript, HTML et CSS. De plus, je suis convaincu d'avoir acquis une plus grande autonomie, car j'ai su m'adapter avec succès à de nouvelles connaissances en apprenant par moi-même. (Maël)

VI.PERSPECTIVES

Malgré l'aboutissement de notre projet nous avons quand même certains points que nous aurions aimé réaliser ou que nous souhaitons réaliser dans le futur. Tout d'abord nous aurions aimé régler si possible le léger problème de calibration qu'on a sur l'angle y. Nous pensions aussi à améliorer le visuel du site et peut être rajouté des options tel qu'une horloge ou une rubrique documentation sur les capteurs inertielle. Enfin nous aurions bien aimé modéliser la carte avec tous ces composants pour avoir un rendu plus réaliste.

VII.BIBLIOGRAPHIE

https://youtu.be/Ihp_cG7c2Rk
https://youtu.be/lvxqvNXniVc
https://flask.palletsprojects.com/en/2.3.x/
https://www.python.org/
https://www.movella.com/products/xsens?utm_feeditemid=&utm_device=c&utm_term=xsens&utm_source=google&utm_medium=ppc&utm_campaign=&hsa_cam=15264159507&hsa_grp=133365416361&hsa_mt=e&hsa_src=g&hsa_ad=561667538299&hsa_acc=1306794700&hsa_net=adwords&hsa_kw=xsens&hsa_tgt=aud-1068507372166:kwd-312964997959&hsa_ver=3&utm_feeditemid=&utm_device=c&utm_term=xsens&utm_source=google&utm_medium=ppc&utm_campaign=Brand+%7C+All+%7C+Search&hsa_cam=15264159507&hsa_grp=133365416361&hsa_mt=e&hsa_src=g&hsa_ad=561667538299&hsa_acc=1306794700&hsa_net=adwords&hsa_kw=xsens&hsa_tgt=aud-1068507372166:kwd-312964997959&hsa_ver=3&gclid=CjwKCAjwp6CkBhB_EiwAlQVyxaQdn985GDZsRFSsnNZnmSKWb1sqKurDyUU-u0cEvNaivFmOwLXlRhoCh1AQAvD_BwE