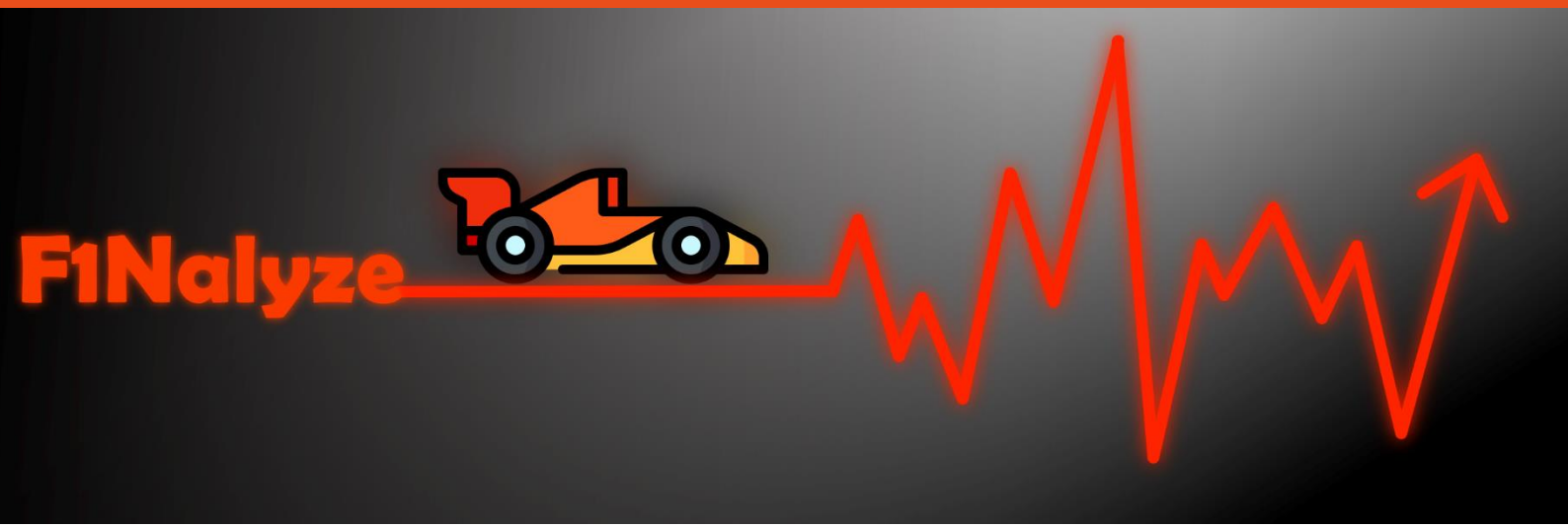
LES TROPHÉES NSI

Édition 2023

DOSSIER DE CANDIDATURE
PRÉSENTATION DU PROJET



Ce document est l'un des livrables à fournir lors du dépôt de votre projet : 4 pages maximum (hors documentation).

Pour accéder à la liste complète des éléments à fournir, consultez la page **Préparer votre participation**.

Vous avez des questions sur le concours ? Vous souhaitez des informations complémentaires pour déposer un projet ? Contactez-nous à **info@trophees-nsi.fr**.

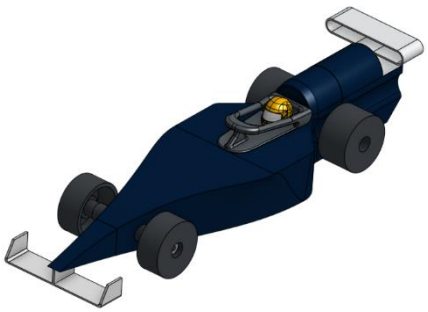
NOM DU PROJET : F1Nalyze

> PRÉSENTATION GÉNÉRALE :

L'idée du projet est de récolter les accélérations d'une carte Arduino Nano 33 BLE Sense (voir figure 4) via une connexion Bluetooth Low Energy (BLE). Ces accélérations sont alors converties en vitesse instantanée puis en trajectoires. L'objectif finale étant d'attacher cette carte à une voiture miniature pour obtenir les grandeurs citées précédemment.

A l'origine, l'intérêt du projet est d'aider les élèves de terminale de section européenne en spécialité science de l'ingénieur. Ces derniers participent au concours « F1 in school ». Ils doivent construire une formule 1 miniature propulsée par une cartouche de gaz et faire la course contre d'autres voitures du même type.

Notre programme peut ainsi les aider à mesurer précisément la vitesse de pointe et les accélérations de la formule 1 pour potentiellement déceler un problème et s'améliorer.



Modèle 3D de la mini formule 1

> ORGANISATION DU TRAVAIL :

Notre équipe est composée de trois élèves :

- Inaki, qui s'occupe des programmes de l'arduino pour la récupération des données et leur transfert.
- Gautier, qui s'occupe de la conception de l'interface graphique en python (voir figure 1)
- Romain, qui s'occupe de calculer et représenter graphiquement les vitesses et les trajectoires (voir figure 2)

Nous travaillions principalement pendant les cours de NSI, alors que les autres élèves travaillaient sur un autre projet. Nous nous sommes également réunis plusieurs fois en dehors de ces cours pour résoudre des problèmes, mettre en commun nos programmes ou encore pour tester l'efficacité de notre programme.

Mais nous avons également travaillé chez nous et pour bien communiqué en-dehors des cours, nous avons créé un serveur Discord dédié à ce projet et un dossier commun dans Google Drive. Pour coder, nous avons utilisé d'une part Visual Code Studio et

EduPython pour le code python et d'autre part Arduino IDE pour le programme arduino.

LES ÉTAPES DU PROJET :

Durant le développement de notre projet, nous avons eut le temps de réfléchir à d'autres fonctionnalités ou méthodes pour notre programme.
Voici l'évolution de notre travail :

1. Nous avons commencé par créer chacun de notre côté l'interface (pour Gautier), la méthode permettant d'aboutir à des vitesses puis des trajectoires (pour Romain) et le script Arduino permettant la connexion Bluetooth entre l'ordinateur et la carte (pour Inaki).

2. Puis, nous avons mis en commun ces trois parties.

3. Nous nous sommes ensuite rendu compte que la méthode de transmission des accélérations en direct était trop lente pour obtenir un résultat fiable (voir figure 3). Nous avons donc mis au point une nouvelle méthode d'acquisition : il s'agit de stocker les accélérations dans la mémoire vive de la carte Arduino puis de les transmettre, toujours pas Bluetooth Low Energy.

4. Puis, nous avons changé la méthode de transmission BLE.

5. Et enfin, nous avons ajusté les calculs des vitesses et des positions.

> FONCTIONNEMENT ET OPÉRATIONNALITÉ :

Le projet est terminé dans sa totalité, même si quelques bugs dus aux boucles asynchrones ou à la connexion Bluetooth peuvent survenir. De plus, il peut y avoir une dérive des mesures, la carte arduino mesure alors une petite accélération alors qu'elle est au repos, ce qui peut fausser plus ou moins les vitesses et les positions. En plus de cela, nous avons rencontré plusieurs difficultés lors du développement de notre programme :

- Il n'existe pas de graphiques matplotlib pouvant évoluer au cours du temps. Il a donc fallu créer une class pouvant faire cela.

- Pour établir un connexion Bluetooth, il faut lancer le programme dans un thread différent. Cependant les threads ne peuvent être lancé qu'une fois. Il a donc fallu

créer une class héritant de la class threading.Thread pouvant être lancé plusieurs fois.

- La connexion Bluetooth était trop lente et trop lente et peu fiable (certaines accélérations n'étaient pas reçues par l'ordinateur ce qui pouvait entrainer des problèmes pour traiter toutes les autres) nous avons donc décidés d'envoyer les données sous formes de String, pour réduire le temps de transmission et plus de précision (même si une donnée n'est pas réceptionnée, cela n'a aucun impactes sur les autres) .

- Il existe peu de documentation sur la connexion Bluetooth des cartes arduino.

- La plupart des librairies pour communication entre pc et arduino sont faites pour linux

- Le stockage des données était difficile à mettre en place.

> OUVERTURE :

Nous sommes fiers d'avoir pu terminer notre projet et surtout de pouvoir apporter notre aide à nos camarades. Il reste cependant quelques améliorations à faire, en particulier au niveau de la précision des accélérations : pour être sûre que les accélérations soit parfaitement juste, il faudrait prendre en compte l'orientation de la carte, et donc utiliser le gyroscope intégré à la carte. De plus, il faudrait trouver un moyen de fixer la carte arduino de telle manière à ce qu'elle n'est aucun impacte sur l'aérodynamique de la mini formule 1.

DOCUMENTATION

Mise en place du programme :

Pour que le programme fonctionne correctement, il faut :

• Mettre le fichier "Logo" dans le même répertoire que le programme python (ou bien supprimer la ligne 108 du programme python)
• Créer des sous-dossiers "Données" et "Graphiques" (avec la même orthographe, sinon modifier les lignes avec le nom de dossier 532, 743 et 733)

Pour téléverser le programme dans la carte arduino, il faut :

• Télécharger l'application d'Arduino à l'adresse suivante :
https://www.arduino.cc/en/software/
• Ouvrir le fichier "ScriptArduino.ino" avec cette application (le programme doit être dans un dossier du même nom)
• Installer les librairies demandées : "ArduinoBLE.h" et "Arduino_LSM9DS1.h"
• Bien sélectionner le port de la carte dans le menu "tools" et enfin cliquer sur la flèche en haut à gauche

Pour changer le nombre ou le temps d'acquisition (à faire dans le programme arduino) :

• Changer à la ligne 16, 17, 18, 82 et 114 le nombre de valeurs d'accélérations voulus (attention, l'accéléromètre de la carte peut monter maximum jusqu'à 1400 Hz)
• Changer à la ligne 88 le délai entre 2 acquisitions (en millisecondes)
•Finalement, changer le nombre de secondes d'attente du programme python à la ligne 324
(Pour obtenir le temps d'acquisition en seconde, il faut multiplier le nombre de valeurs avecle délai entre 2 acquisitions puis le diviser par 6)

Guide d'utilisation :

Avant toute chose, il faut impérativement s'assurer que le Bluetooth de l'ordinateur soit actif.
Il ne faut pas se connecter directement à la carte mais juste le laisser allumer.

Marche à suivre pour acquérir les accélérations de la carte arduino :

1. Choisir la façon d'acquérir en cliquant sur le bouton correspondant à gauche. • Si c'est la première acquisition, entrer l'adresse MAC de la carte arduino.

2. Poser la carte et appuyer sur « Configurer les offsets » (après cette étape ne plus pivoter la carte pour éviter que l'influence de la gravité ne fausse les mesures)

3. Lorsque l'on est prêt, appuyer sur « Démarrer » pour débuter l'acquisition.
   Si l'on a choisi la méthode indirecte, attendre que le programme ait fini (à partir du moment où les courbes sont affichées).
   Sinon, appuyer sur le bouton « arrêter » au moment voulu pour stopper l'acquisition.

Voici un tableau récapitulatif des actions possibles :

Action	**Description**	**Commande**
Analyser	Permet d'accéder aux vitesses et au positions de la carte arduino.	Cliquer sur le bouton analyser
Enregistrer une image	Enregistre une image .png du graphique voulu	Dans le menu file, aller dans « Sauvegarder une image » puis cliquer sur un graphique. Enfin entrer le nom du fichier sans extension
Enregistrer des données	Enregistre un fichier .csv contenant toutes les accélérations et leur temps	Dans le menu file, aller dans « Sauvegarder des données » puis cliquer sur un graphique. Entrer le nom du fichier sans extension.

Visualiser des données enregistrées	Affiche sur un graphique le fichier sélectionné	Ctrl + O ou bien dans file, cliquer sur "Afficher des données". Puis entrer le nom du fichier .csv (sans extension et le fichier doit être présent dans le fichier « Données »)
Changer de carte arduino	Change l'adresse MAC recherché par le programme	Dans edit, cliquer sur "Changer d'appareil" ou bien appuyer sur Ctrl + D. Puis entrer l'adresse MAC de la nouvelle carte (attention si vous quittez cette fenêtre avant d'avoir entré une nouvelle adresse, l'ancienne adresse sera quand même supprimée).
Changer la couleur d'un graphique	Change la couleur de la courbe d'un graphique	Dans « edit », aller sur « Changer la couleur d'un graphique », choisir le graphique puis la couleur.
Quitter	Ferme le programme	Touche Echap
Ouvrir une nouvelle fenêtre	Ferme la fenêtre et en ouvre une nouvelle (en cas de bug)	Ctrl + N ou bien dans file cliquer sur « Nouvelle fenêtre »
Nettoyer les graphiques	Enlève toutes les courbes des graphiques	Ctrl + R ou bien dans edit , appuyer sur « refresh »
Afficher en 3D (Uniquement disponible dans la fenêtre analyser)	Affiche un graphique 3D représentant la trajectoire de la carte arduino	Dans file, cliquer sur « Visualiser en 3D »

Spécification technique :

Notre projet est décomposé en 2 programmes, ils communiquent entre eux via Bluetooth Low Energy (BLE). Ainsi, il y a un programme en python (sur l'ordinateur):

Architecture : orienté objet

Bibliothèques utilisées :

- *asyncio → indispensable à notre bibliothèque bleak*
- *time → permet d'avoir accès au temps réel*
- *threading → sert à exécuter plusieurs threads en même temps*
- *tkinter → pour la création de l'interface*
- *warnings (simplefilter) → permet de ne pas afficher les messages d'erreurs*
- *bleak (BleakClient, BleakError) → pour la connexion Bluetooth*
- *matplotlib (pyplot, Figure, FigureCanvasTkAgg, NavigationToolbar2Tk, style) → pour afficher des graphiques, autant sur l'interface tkinter qu'avec pyplot*

Utilisation : Gère l'interface et la récupération des données envoyées par la carte arduino

L'autre programme est un programme en arduino (sur la carte arduino), qui se rapproche syntaxiquement d'un langage C simplifié:

Architecture : orienté objet

Bibliothèques utilisées :

- *ArduinoBLE.h → indispensable pour communiquer en BLE*
- *Arduino_LSM9DS1.h → sert à avoir accès à l'accéléromètre*

Utilisation : mesure l'accélération de la carte en g puis les transferts via BLE

<u>Unités :</u>

Les différentes unités ne sont pas indiquées sur les différents graphiques, c'est pourquoi nous les précisons ici, à noter que nous utilisons les unités du système international (SI) :

Accélérations : $m.s^{-2}$
Vitesses : $m.s^{-1}$
Distance : m
Temps : s

Pour réussir à obtenir des vitesses et des trajectoires via des accélérations, nous avons du créé nos propres formules, voici leur justification :

Formules pour convertir une accélération en vitesse :

On a : $\boldsymbol{a} = \frac{\Delta \boldsymbol{v}}{\Delta \boldsymbol{t}}$

⇔ $\Delta v = a \times \Delta t$

Or **$\Delta v = v_{suivant} - v_{précédant}$** , ainsi :

$v_{suivant} - v_{précédant} = a \times \Delta t$

⇔ $v_{suivant} = v_{précédant} + a \times \Delta t$

Formule pour convertir une vitesse en trajectoire :

On a : $\boldsymbol{v} = \frac{\boldsymbol{d}}{\boldsymbol{t}}$

⇔ $d = v \times \Delta t$

On obtient également les mêmes résultats en faisant une étude dimensionnelle :

Comme **$[a] = m.s^{-1}$** et **$[\Delta t] = s$** :

$[a \times \Delta t] = m.s^{-2} \times s$
$= m.s^{-1}$

⇔ [a × Δt] = vitesse

Ainsi la multiplication d'une accélération et d'une durée donne une vitesse . Pour obtenir la vitesse exacte on doit alors ajouter la dernière vitesse. C'est pourquoi :

$v = v_{précédant} + a \times \Delta t$

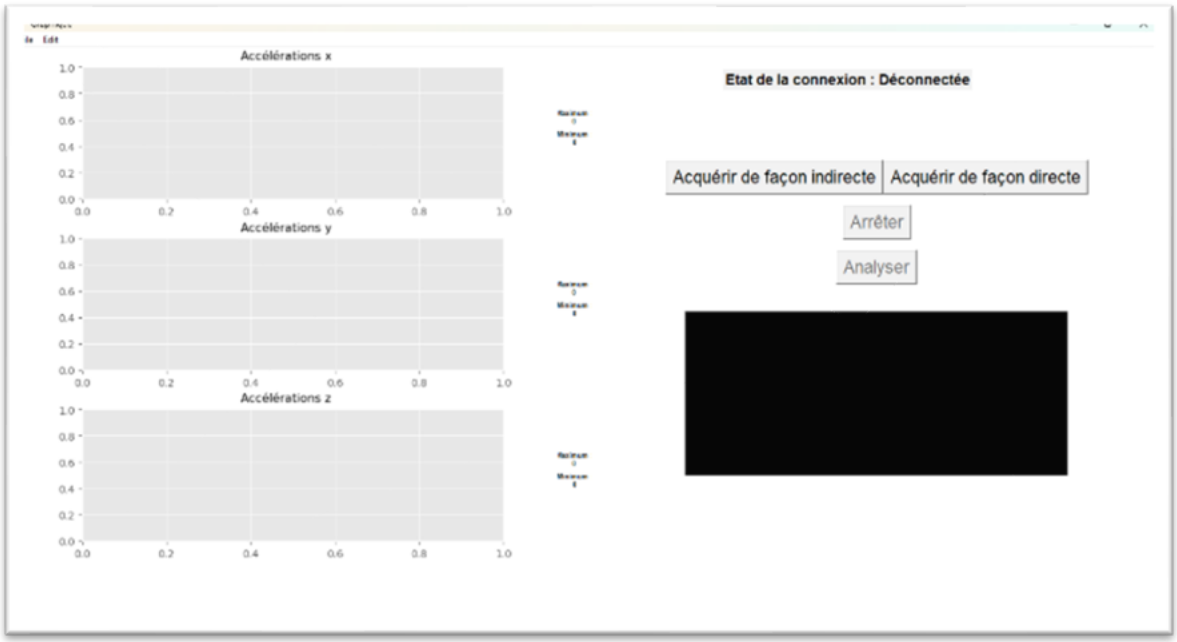
On remarque que cette formule est identique à celle trouvée précédemment, et on peut appliquer le même raisonnement à celle des positions, étant donné que :

$[v \times \Delta t] = m.s^{-1} \times s$
$= m$

⇔ distance

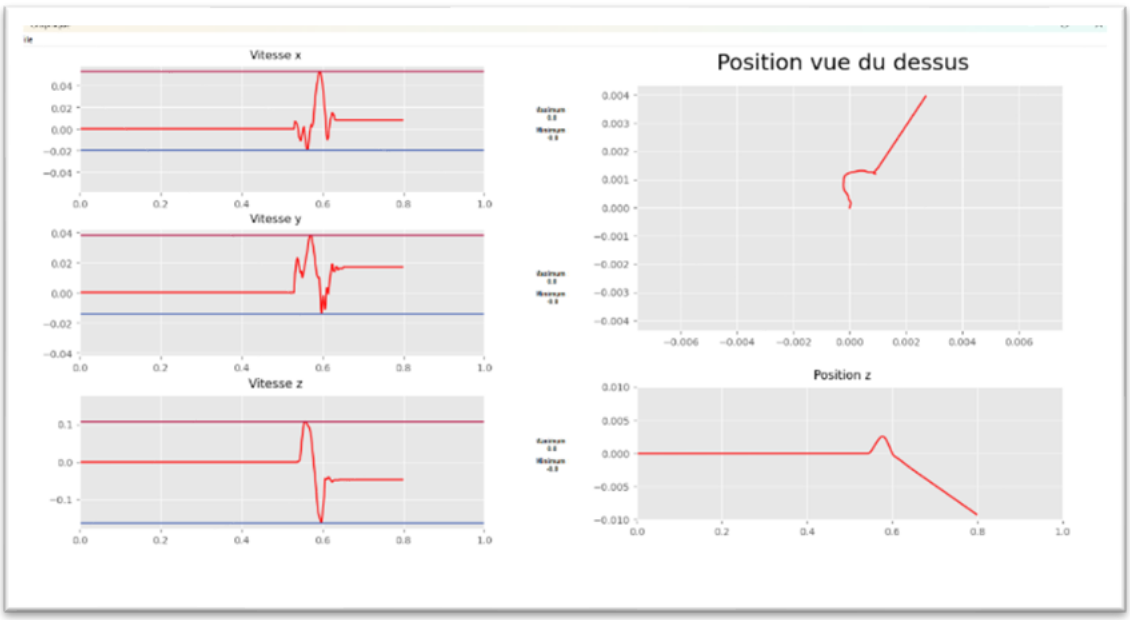
ANNEXE

Figure 1



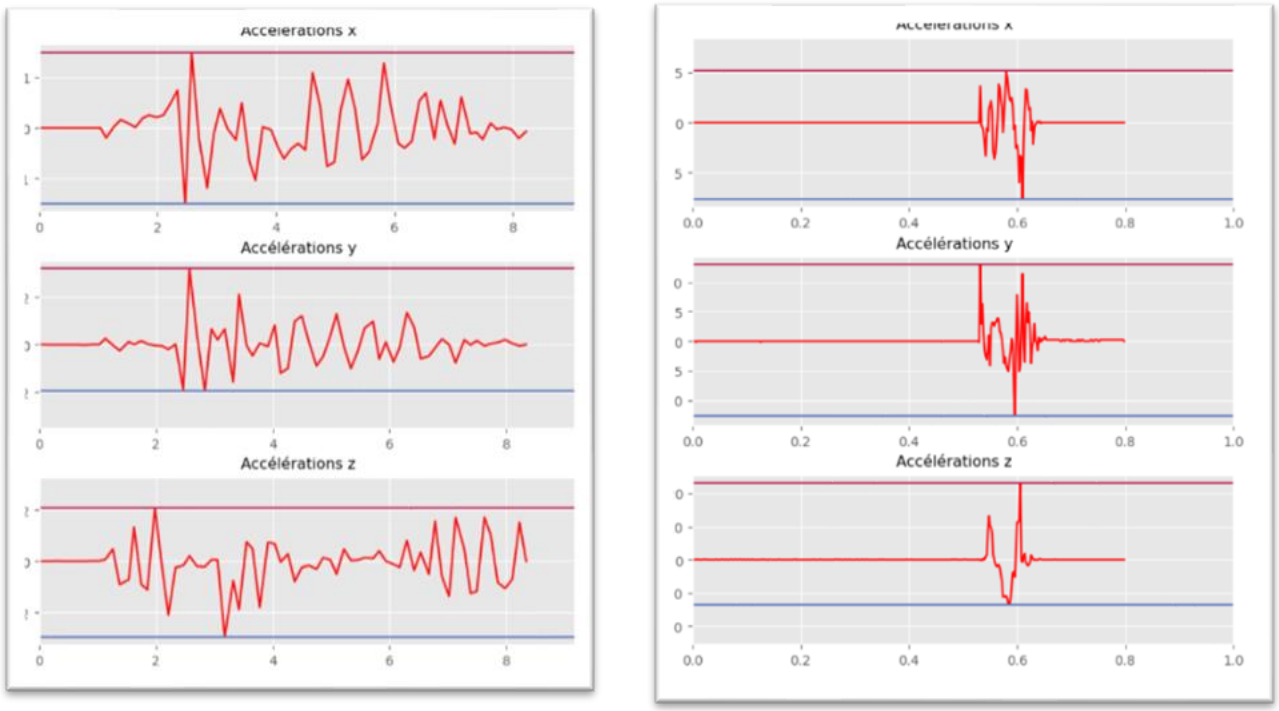
Capture d'écran de l'interface à l'ouverture

Figure 2



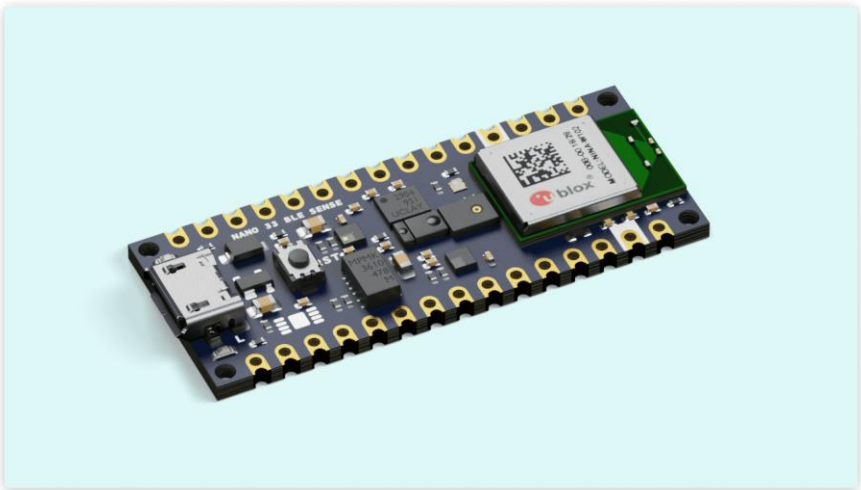
Capture d'écran de la fenêtre d'analyse

Figure 3



Graphiques d'accélérations prises directement (gauche) et indirectement (droite)
les accélérations ne sont pas les mêmes mais on observe que le graphique indirect à beaucoup plus de point sur un laps de temps beaucoup plus court)

Figure 4



Une Arduino nano 33 BLE Sense