# Première version

Afin de récupérer les données des imus, on reprend le code de linorobot pour « linobase ». Des modifications sont nécessaire car il est prévu que pour un seul imu à la fois. Il envoie les messages sous la forme d’un « linorobot\_msgs ».

Les paquets de filtre Madgwick et de filtre de kalman ont été modifier pour y ajouter les topics nécessaires à l’implantation d’une plateforme avec des IMUs à testés. Cependant il est aussi possible de simplement utilisé les commandes de remapping de nom de topics.

# Seconde méthode

Implantation d’un filtre ZVU (Zero Velocity Update) vu dans cet exemple :

<https://github.com/Wojtek120/IMU-velocity-and-displacement-measurements#zvu>

Une première tentative a été faite avec la création d’un paquet personnaliser implémentant ce filtre. Cependant, la complexité fait qu’il est plus simple de développer le code avec Jupiter lab en python.

Un inconvénient du filtre est qu’il n’est pas temps réels.

# Troisième méthode

Le kalman

<https://www.kalmanfilter.net/alphabeta.html>

<https://github.com/rlabbe/Kalman-and-Bayesian-Filters-in-Python>

<https://scholarworks.calstate.edu/downloads/dv13zt241>

## Collecte des données de test

Les données pour tester les filtres de kalman sur notebook jupyter sont collectées en enregistrant le topic « /testPlatform\_imu/data » dans un rosbag.

% rosbag record /testPlatform\_imu/data –O file\_name.bag

Transfert les données dans un fichier texte :

% rostopic echo –b file\_name.bag –p /topic\_name > file\_name.txt

Convertie en .csv via un convertisseur en ligne.

Ouvrir le fichier dans Excel :

* Sélectionner la colonne A
* Onglet Donnée -> convertir
* Délimiter
* Séparateur virgule
* Terminer

Sélectionner à nouveau la colonne A qui devrait ne contenir le timestamp des données

* Accueil -> dérouler l’encadrer marquer « standard » -> Autres format numérique
* Nombre -> nombre de décimal : 0 -> OK

# Quatrième méthode : RIDI

<https://yanhangpublic.github.io/ridi/index.html>