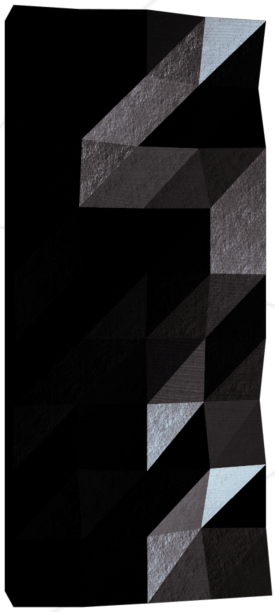


ft_kalman

Plus d'outils pour trouver charly

Tiago Lernould (tlernoul),
Eliott Suits (esuits),
Damien Cuvillier (dacuvill)

Summary: *Pour ce projet, vous aurez besoin de créer un filtre de kalman pour suivre une trajectoire à partir d'informations faussées et incomplètes*

Contents

Chapter I

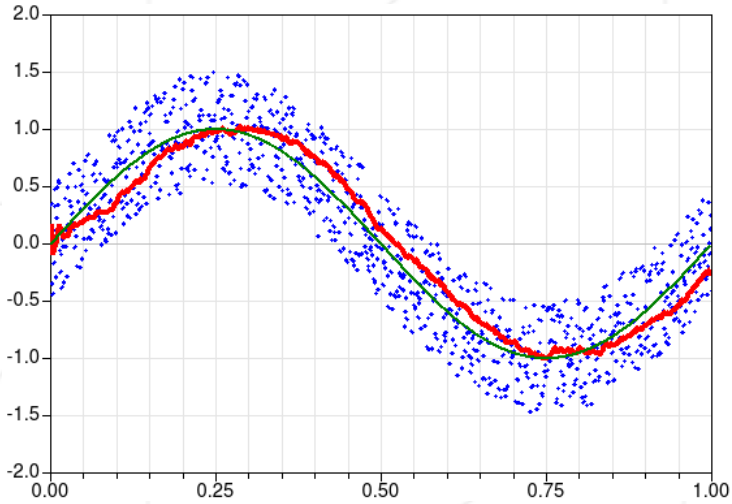
Préambule

The missile knows where it is at all times.
It knows this because it knows where it isn't.
By subtracting where it is from where it isn't, or where it isn't from where it is (whichever is greater), it obtains a difference, or deviation.
The guidance subsystem uses deviations to generate corrective commands to drive the missile from a position where it is to a position where it isn't, and arriving at a position where it wasn't, it now is.
Consequently, the position where it is, is now the position that it wasn't, and it follows that the position that it was, is now the position that it isn't.
In the event that the position that it is in is not the position that it wasn't, the system has acquired a variation, the variation being the difference between where the missile is, and where it wasn't.
If variation is considered to be a significant factor, it too may be corrected by the GEA. However, the missile must also know where it was. The missile guidance computer scenario works as follows.
Because a variation has modified some of the information the missile has obtained, it is not sure just where it is. However, it is sure where it isn't, within reason, and it knows where it was.
It now subtracts where it should be from where it wasn't, or vice-versa, and by differentiating this from the algebraic sum of where it shouldn't be, and where it was, it is able to obtain the deviation and its variation, which is called error.

Chapter II

Introduction

Le but de ce projet est de créer un filtre de kalman, un algorithme utilisé en navigation (dans les airs, sur terre, dans l'espace, en bref partout), en économie, et même en vision par ordinateur!
Vous allez découvrir des concepts tels que les matrices de covariance, la théorie du signal, et d'autres choses entre deux...
Il est vivement recommandé d'avoir terminé le projet matrix avant de commencer celui-ci.



Vous êtes la centrale inertielle (imu/Inertial Measurement Unit) d'un véhicule générique (ni une voiture, ni un avion ou un bateau) se déplaçant dans un environnement générique (sans air et gravité).
Les seules informations que vous avez sont les suivantes:

- Votre vraie position initiale (X,Y,Z en mètres, au départ),
- Votre vraie vitesse initiale (en km/h, au départ),
- Votre accélération actuelle (en m/s², chaque centième de seconde),
- Votre direction actuelle (en angles d'Euler),
- Votre position GPS actuelle (X,Y,Z en mètres, toutes les trois secondes),

Mais, étant donné que nous vivons dans un monde imparfait, certaines de ces mesures sont affectées par un bruit blanc gaussien:

- Accéléromètre : $\sigma = 10^{-3}, \upsilon = 0$
- Gyroscope : $\sigma = 10^{-2}, \upsilon = 0$
- GPS : $\sigma = 10^{-1}, \upsilon = 0$

Chapter III

Objectifs

Vous trouverez dans les fichiers du projet un programme ILC (CLI) qui attendra vos signaux sur un port UDP et sera votre source d'informations.

Après s'être connecté avec le programme, vous devrez lui envoyer un signal qui lancera la génération de la trajectoire.
Passé cette étape, vous recevrez vos premières informations nécessaires pour initialiser votre filtre de Kalman et le programme attendra votre première estimation.

Format requis pour les estimations ("X Y Z")

```
1.7325073314060224 -2.2213777837034083 0.49999962025821726
```

Ainsi commencera un dialogue entre votre filtre et le programme de flux de mesures, qui attendra votre estimation de position avant de renvoyer des données actualisées, et ce jusqu'à la fin du parcours. Evidemment, votre estimation de position doit être proche des coordonnées réelles. Si votre estimation est éloignée de plus de 5 mètres de la position réelle, ou si votre filtre met plus d'une seconde à répondre, alors le programme de flux de mesures enverra une erreur et arrêtera la communication.

Pour ce qui est du reste des informations, vous devrez chercher par vous-mêmes! (-h)

```
./imu-sensor-stream -s 42 -d 42 -p 4242
```

Chapter IV

Consignes générales

- Vous devez implémenter un filtre de Kalman sans machine-learning ou d'autres algorithmes.
- Votre code doit être en C, C++ (standard C++17 ou moins) ou en Rust
- Votre code doit compiler statiquement
- Vous ne pouvez évidemment pas utiliser une librarie opérant le filtrage à votre place, les librairies de maths sont cependant autorisées
- Vos fonctions ne doivent pas s'arrêter inopinément (segmentation fault, bus error, double free, panic, etc) mis à part dans des cas non définis par l'énoncé.
- Toute mémoire allouée doit être proprement libérée. Aucune fuite de mémoire ne sera tolérée.
- Si le langage le requiert, vous devez joindre un makefile qui compilera vos fichiers source avec les flags -Wall, -Wextra and -Werror, et celui-ci ne devra pas relink.

Chapter V

Partie obligatoire

Votre filtre doit toujours fonctionner avec des trajectoires de durées inférieures ou égales à 90 minutes avec la quantité de bruit par défaut.

`Les points suivants ne s'appliquent qu'à des parcours de 90 minutes ou moins. Rien ne sera considéré passé cette durée.`

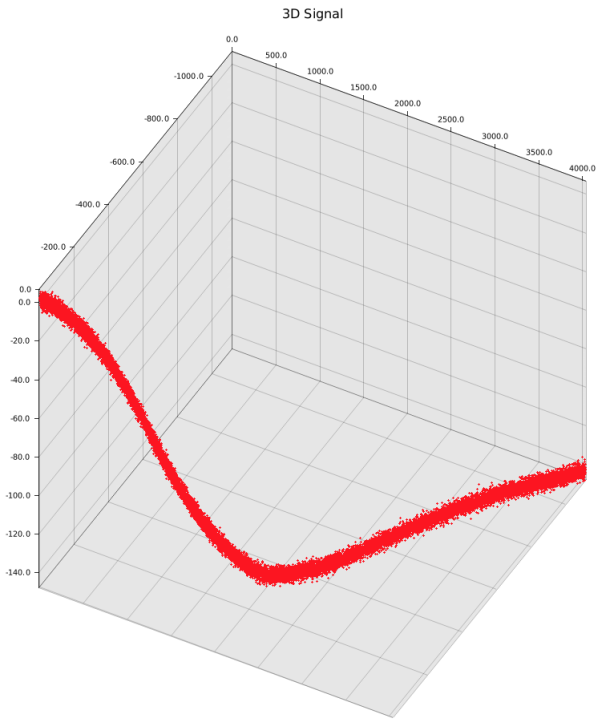
Votre programme doit être décemment optimisé et donc ne jamais timeout, et si cela doit être le cas le programme devra s'arrêter proprement (pas de crash, segfault ou fuite de mémoire).

Aussi, votre filtre devra être assez précis pour ne jamais envoyer une estimation trop écartée de la réalité, et si cela devait arriver il devra s'arrêter proprement également.

Chapter VI

Partie bonus

- Un visualisateur de trajectoire (d'un tracé basique en 2d à un visualisateur 3d avec affichage de la variance).
- Un filtre très rapide répondant en quelques millisecondes en moyenne (voir l'argument "meanspeed" pour estimer cela).
- Un filtrage très précis pouvant gérer plus de bruit que la quantité par défaut (voir l'argument "noise").
- Une nouvelle fonctionnalité à laquelle nous n'avons pas pensé (un ajout purement esthétique ne comptera pas).



Peut-être est-ce un bon exemple de visualisateur de trajectoire en 3d.

Ou pas...Qui sait.

Chapter VII

Rendu et peer-évaluation

Nous savons qu'il s'agit d'un projet très technique et complexe, c'est pourquoi vous devrez absolument assimiler le concept du filtre de Kalman et son fonctionnement, et parce que cet aspect sera extrêmement important pendant votre évaluation.

Rendez votre travail sur votre dépot GiT comme d'habitude. Seul le travail présent sur votre dépot sera évalué en soutenance.