Aabu

Zeste de savoir

Déterminer l'inclinaison d'une pyboard en MicroPython

14 décembre 2022

Table des matières

Introduction

La pyboard comporte un petit accéléromètre, ce qui permet de faire une estimation de l'inclinaison de la carte!

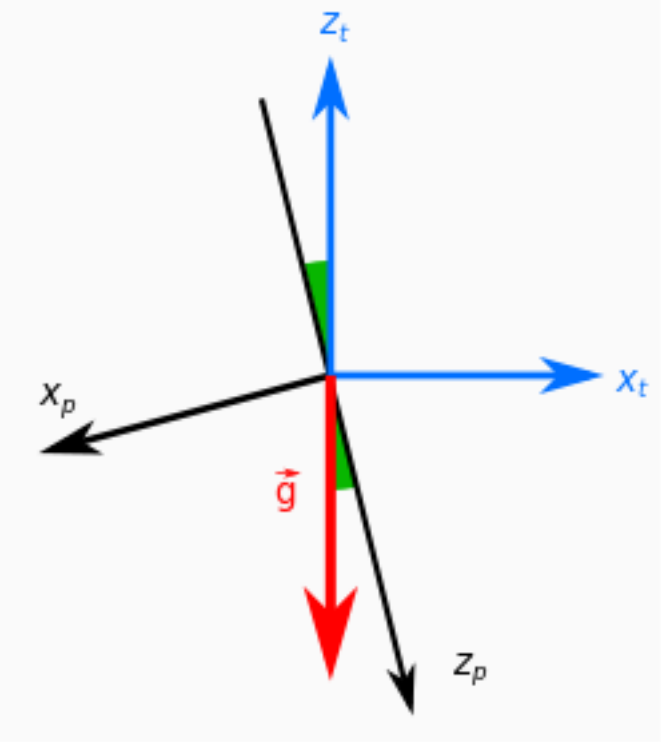
1. Principe

Comme son nom le suggère, un accéléromètre mesure une accélération. Celui intégré à la pyboard fournit une valeur d'accélération pour chacun des trois axes x, y et z, qui donnent ainsi les coordonnées d'un vecteur accélération dans un repère associé à la pyboard.

Quand la pyboard est au repos ou presque au repos, l'accélération mesurée correspond l'accélération due à la gravitation terrestre, $\vec{g}$. Autrement dit, il s'agit des coordonnées du vecteur $\vec{g}$ dans le repère associé à la pyboard.

Pour un projet, j'ai besoin de connaître l'inclinaison de la carte par rapport à la verticale. Comme $\vec{g}$ est vertical dans le repère terrestre, l'angle entre $\vec{g}$ et l'axe z dans le repère associé à la pyboard correspond à l'angle entre la verticale et la pyboard.

Sur le schéma ci-dessous, cela se traduit par le fait que les deux angles verts sont identiques (à condition de faire attention à l'orientation).



On peut retrouver l'angle à partir des coordonnées avec un peu trigonométrie. En restreignant l'angle à [-90°; +90°], il suffit de déterminer le sinus de l'angle pour obtenir l'angle avec sa fonction réciproque arcsin.

L'angle recherché est alors:

$$\theta = \arcsin \frac{x}{\sqrt{x^2 + z^2}}$$

2. Implémentation

L'implémentation est une simple transcripton de la formule mathématique, en récupérant les coordonnées de l'accéléromètre. J'ai hésité à faire un retour uniquement en degrés, ce qui s'est traduit par un non-choix: il y a une option pour renvoyer des radians.

```
1  import pyb
2
3  class AngleEstimator:
4      def __init__(self):
5          self.accel = pyb.Accel()
6
7      def angle(self, unit='deg'):
8          x = self.accel.x()
9          z = self.accel.z()
10         sin = x / math.sqrt(x * x + z * z)
11         angle = math.asin(sin)
```

```
12          if unit == 'deg':
13              return angle * 180 / math.pi
14          else:
15              return angle
```

3. Performance

3.1. Vitesse d'exécution

La méthode `angle()` s'exécute en environ 275 µs, ce qui est assez rapide pour mon usage. Il y a peu à gagner sur les calculs mathématiques, la grosse majorité du temps est prise par les appels à l'accéléromètre (environ 250 µs).

3.2. Résolution/Précision

La précision est assez faible. Des techniques de filtrages peuvent être utiles pour y pallier, mais la méthode peut dépendre beaucoup de l'usage qu'on fait de la valeur d'inclinaison. Aussi, je ne m'étendrai pas dessus.

Cette faible résolution est due à la résolution de l'accéléromètre, qui est très limitée. Le composant est fait pour mesurer des positions grossières, à destination des appareils portables (par exemple pour détecter l'orientation d'un téléphone).

Pour mon besoin, ce paramètre risque d'être limitant, aussi, je devrais peut-être me tourner vers des composants dédiés à ce genre d'application (type IMU).

3.3. Offset

L'offset est un décalage propre à chaque composant et qui devrait être corrigé pour exploiter au mieux le composant. C'est faisable en logiciel ou directement dans le matériel.

L'accéléromètre est un MMA7660FC dispose de pas mal de documentation, notamment pour expliquer comment régler l'offset (*Application Note AN3841*). Curieusement, cette méthode n'est pas présente dans la documentation principale.

Conclusion

Miniature du billet: logo de MicroPython (source).