

Un exemple d'installation d'un système de navigation complet basé sur un PI 5 / OpenCPN / Pypilot.

Quelques prés requis :

Pourquoi le choix d'un PI 5 ?

Ce raspberry est largement suffisant pour faire fonctionner de façon fluide et confortable OpenCPN avec ici une cartographie mondiale CM93 et Pypilot.

Pourquoi ne pas utiliser OpenPlotter ?

Je ne suis pas passé volontairement par les distributions OpenPlotter qui me semblent opaques et j'ai préféré paramétrer et installer moi même tous les logiciels nécessaires au bon fonctionnement de tout ce système afin d'en maîtriser au mieux tous les paramètres. **Voir articles précédents concernant l'installation et configuration sur un PI 5 OS BookWorm avec tous les prés requis d'installation et configuration.**

Pourquoi ne pas utiliser SignalK ?

Je ne suis pas passé par signal K qui permet d'agréger toutes les sources de communication car cela rajoute une couche de communication supplémentaire et donc de la complexité; inutile ici, je veux un système rustique et simple avec des communication directes vers le Raspberry PI 5

Pourquoi NMEA 0183 entre les capteurs et le raspberry

Car c'est un protocole de base très simple utilisé sur des liaisons de type série qui a en plus le mérite d'être en « clair ». Ici même pas de multiplexeur, juste des câbles série avec des adaptateurs RS232 ou RS422 / TTL / USB arrivant sur un hub usb rattaché au raspberry. De plus c'est le protocole de base utilisé directement par le service Pypilot.

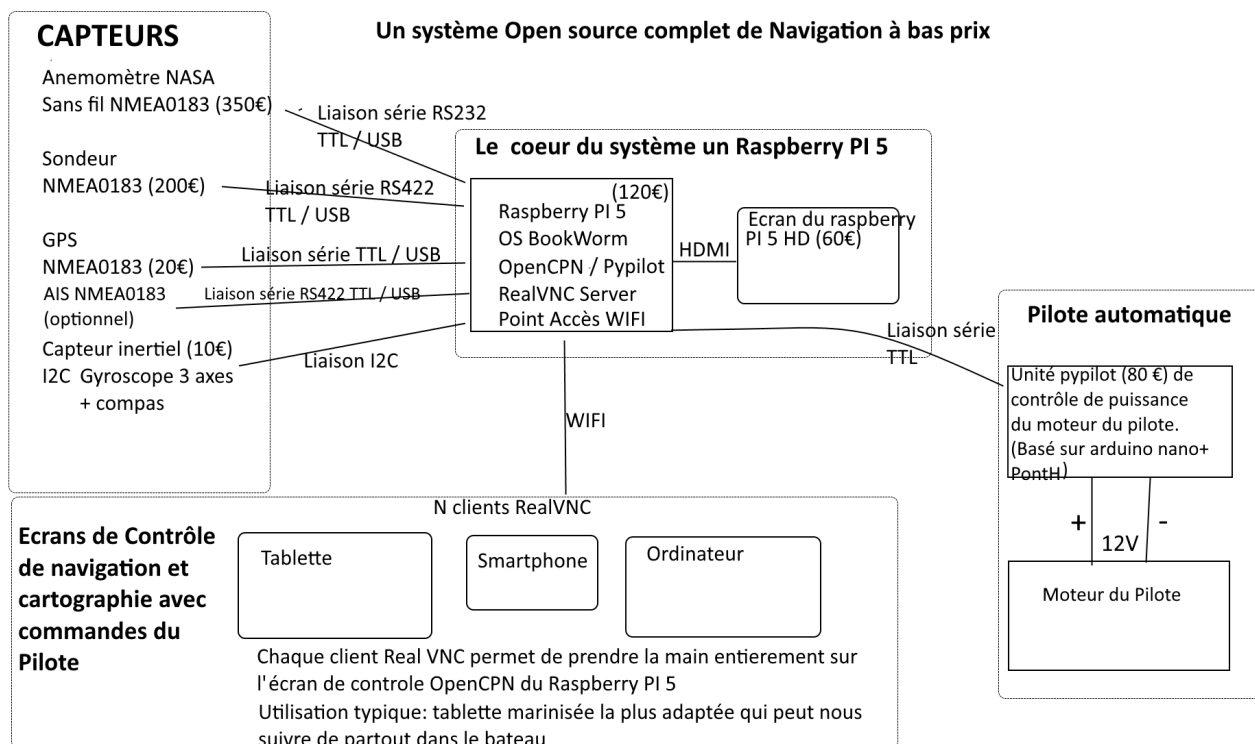
Bref je veux un système le plus simple possible que je serai capable de maîtriser de bout en bout.

Mon beau système B&G avec des écran qui surconsomme fonctionne comme une boîte noire que je ne maîtrise absolument pas en cas de problème, là en installant et configurant tout moi même de A à Z j'espère que que je serai plus autonome, vœux pieux, peut être... ou fantasme du vieux geek que je suis peut être... Mais une chose est sûre je saurai mieux comment fonctionne tout ce bazar et savoir être autonome en navigation en cas de panne est un gros avantage !

Une dernière certitude tout ce système open source consomme très peu (environ 4 fois moins qu'un beau système B&G) et en très longue navigation cela est aussi primordial !

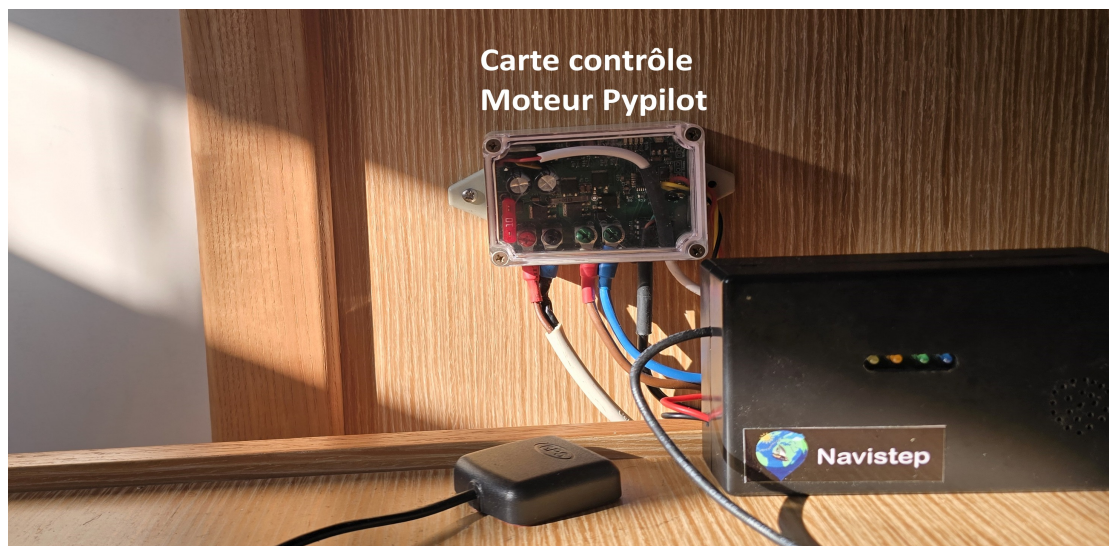
Comme le document précédent ce document s'adresse à des initiés et a des curieux qui veulent utiliser cette solution particulière bâtie autour d'un PI 5 et de ma propre conception de la navigation avec OpenCpn et Pypilot. Il y a certes des options qui consomment moins bâties autour du TinyPilot et d'un raspberry PI 0 W mais je voulais explorer ici une autre voie me semblant ergonomiquement satisfaisante et basée sur un système de navigation complet pour des voiliers de voyage équipés généralement de beaucoup plus de panneaux solaire. Pour des solutions déjà paramétrées, solides et de qualité je vous engage à aller sur le site <https://navitop.fr> qui a le mérite de commercialiser des systèmes bâtis autour de Pypilot. Le concept de Pypilot est tellement riche qu'il a une multitude de possibilités de l'utiliser dans divers systèmes de navigation souvent chacun très personnalisés.... Ceci est donc un exemple de système utilisant OpenCPN et Pypilot parmi tant d'autres.

Schéma de principe :



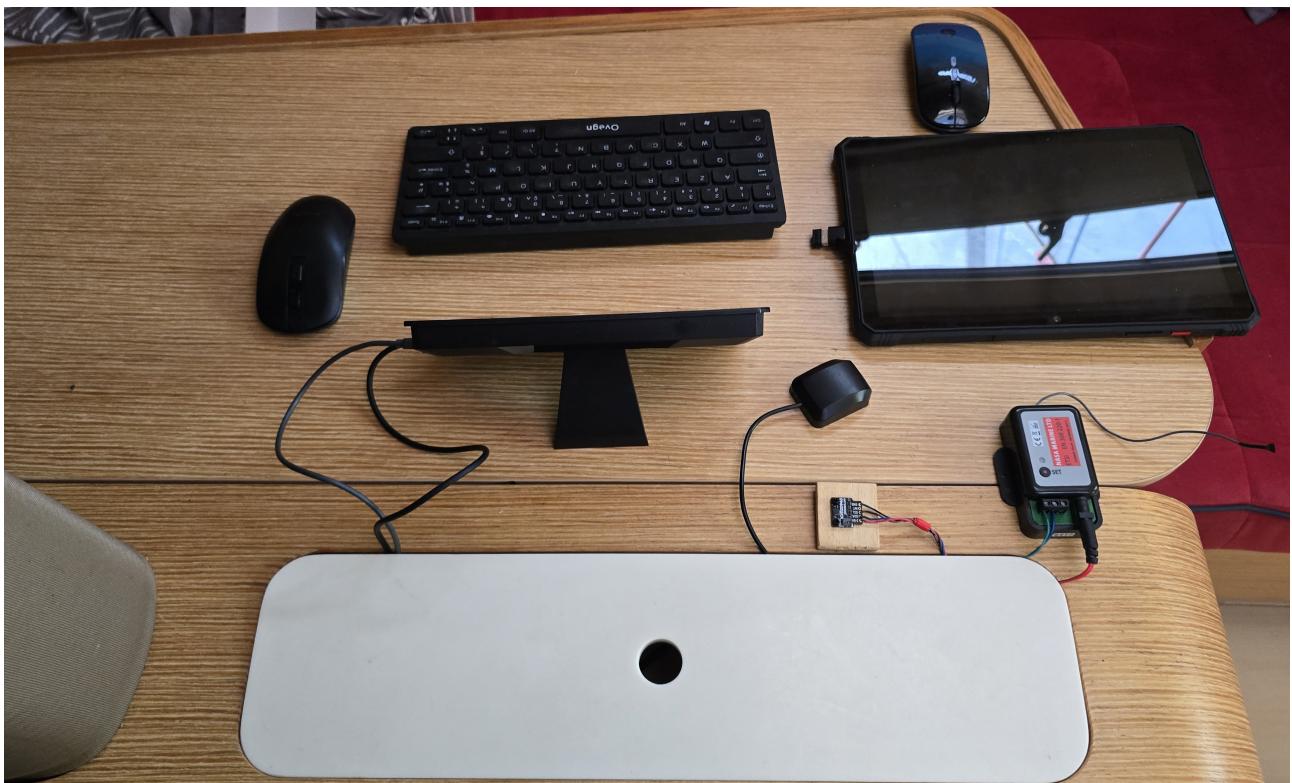
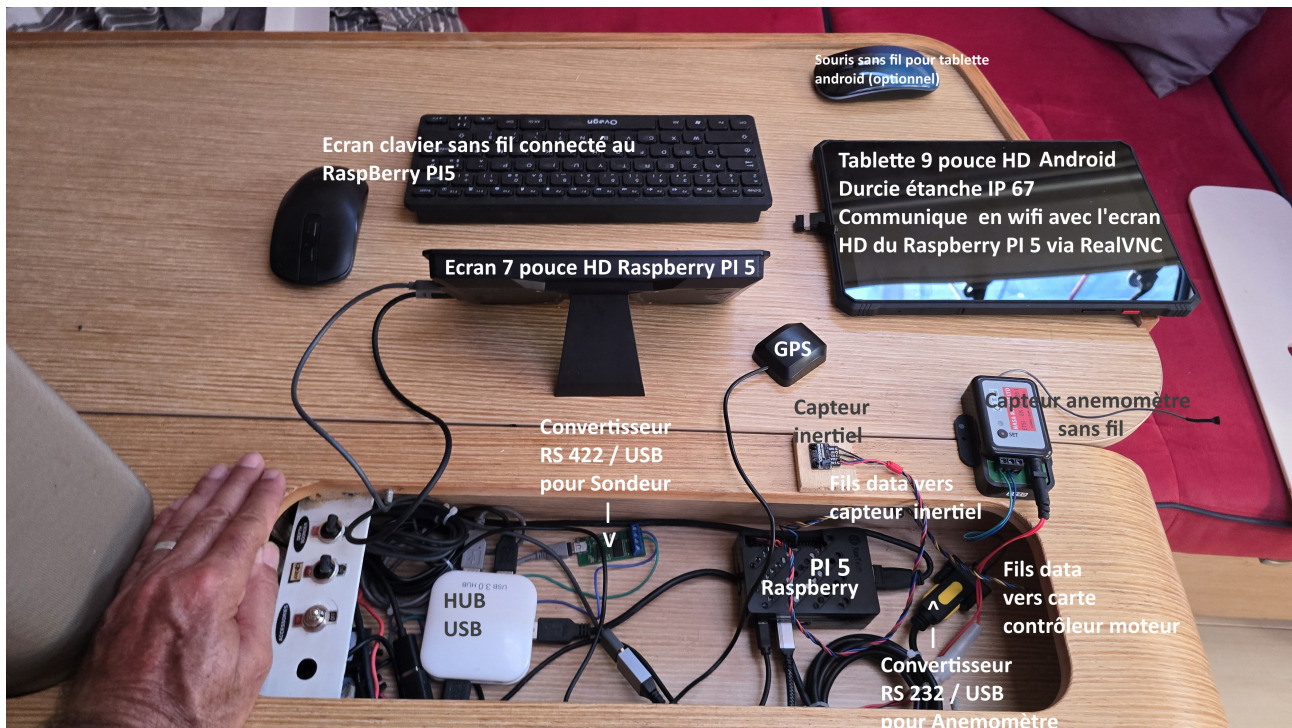
Quelques photos de l'installation

Carte Pypilot de contrôle du moteur du pilote auto :



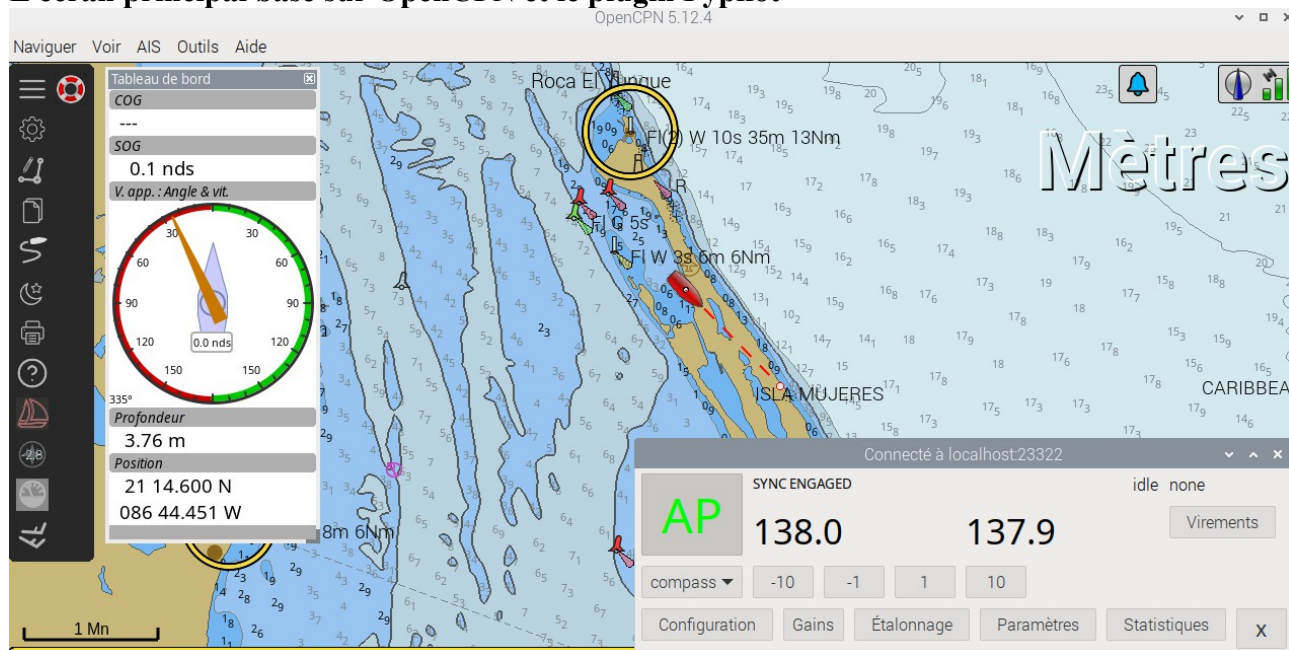
Cette carte Pypilot Open Source de contrôle du moteur du pilote est proche de la table à carte. (Le boîtier Navistep à côté est un tracker que j'ai conçu et est ici hors projet et installation). Cette carte communique via un port série RX TX avec le Raspberry PI 5 qui donne les ordres, elle est alimentée en 12V et via son microcontrôleur arduino nano et un pont H elle transmet ces ordres et fournit la puissance voulue au moteur du pilote.

L'ensemble de tous les autres éléments se trouve dans cet espace restreint de 44cm x 15 cm au milieu de la table du carré.



Hors navigation, une fois rangé, tout est enfermé dans cet espace restreint au milieu de la table (y compris clavier, souris, écran, capteur inertielle, capteur sans fil vent). Plus rien ne traîne !

L'écran principal basé sur OpenCPN et le plugin Pypilot



Sur une cartographie mondiale CM93, on a ici toutes les informations nécessaires utiles à la navigation : COG, SOG, vent apparent force et direction, profondeur, position gps.

Le plugin OpenCpn Pypilot est ici affiché en bas à droite et permet de contrôler le moteur du pilote avec les ordres -1 -10 +10 +1 en mode compas, (il existe aussi un mode vent).

L'ensemble des paramètres du pypilot sont accessibles en dessous avec les boutons Configuration, Gains, Etalonnage, Paramètres, statistiques. *Nous reviendrons dans un prochain document sur l'utilisation des ces paramètres en situation réelle de navigation.*





La prise de main a distance avec une tablette Android et RealVNC

Un écran est directement rattaché au Raspberry PI 5 via un câble HDMI, mais cet écran est aussi accessible a tout autre écran de tablette utilisant ce partage d'écran avec une prise de main a distance via realVNC porté par le point d'accès wifi du Raspberry PI 5 et son adresse IP fixe. Sur l'OS Bookworm RealVNC serveur est pré installé, il suffit d'activer ce service via raspi-config. Sur les postes clients (ici tablette Android) il faut installer RealVNC Client (ici via Google Play sur la tablette android) , logiciel gratuit tant qu'on accède à des adresses IP locales sur un point d'accès wifi local ce qui est le cas ici.

Pour l'adresse IP fixe et le point d'accès wifi du Raspberry PI 5 sur lequel va reposer la connexion RealVNC et la prise de main a distance, aller dans le menu réseau en haut a droite, rechercher l'option Points d'accès et partage de connexion, activez le point d'accès wifi ainsi que le nom de ce réseau (SSID) et le mot de passe associé.

Une procédure détaillée de cette installation avec ses avantages et inconvénients est détaillée dans le document : Prise de main a distance sur le PI 5 BookWorm avec OpenCpn et Pypilot

Configuration dans openCPN:

	Protocole	Entrée/Sortie	Port des données	Statut		
<input checked="" type="checkbox"/>	nmea0183	Entrée	TCP:localhost:20220	✓		
<input checked="" type="checkbox"/>	nmea0183	Entrée	GPSD:localhost:2947	✓		

Dans les paramètres de OpenCPN ne pas déclarer les différents capteurs (anémomètre, GPS, Sondeur, AIS etc...) basés sur le protocole NMEA0183 sur leurs différents ports série existants (du style dev/ttyxxxxx)

Cela se traduira par une absence de données au niveau de OpenCpn. Sur un raspberry et OpenCPN, un port série ne peut être lu par deux processus simultanément, c'est donc le service Pypilot qui lit toute les données à leur source sur les différents ports série et les redistribuent via TCP ou UDP sur des ports distincts. Ici pour tous les capteurs (hors GPS) comme l'anémomètre ou le sondeur le bon paramétrage de connexion est le suivant : **protocole TCP, port 20220 adresse ; localhost**
C'est donc le service Pypilot qui scanne tous ces capteurs et renvoie les data NMEA018 correspondantes via TCP:localhost:20220 à OpenCPN

Pour le GPS il est préférable d'utiliser le service GPSD sur l'adresse localhost et le port 2947

Vérifier que le service gpsd est installé sur l' OS bookworm, si nécessaire l'installer commande :

```
sudo apt install gpsd gpsd-clients python3-gps
```

Pour verifier le fonctionnement du GPS commande : gpsmon (bien vérifier qu'on est en protocole NMEA0183 et nom UBX)

Sur bookworm il est préférable de désactiver le socket lié à gpsd pour le forcer à passer en NMEA018 et non UBX

commandes :

```
sudo systemctl stop gpsd.socket
```

```
sudo systemctl disable gpsd.socket
```

```
sudo systemctl stop gpsd
```

```
sudo systemctl disable gpsd
```

Si le GPS est en mode UBX le forcer a passer en mode NMEA0183, commande :

```
sudo gpsctl -n /dev/ttyACM0
```

(ici ttyACM0 correspond au port usb utilisé par ce GPS sur mon raspberry)

éventuellement lancer gpsd sans auto configuration et en NMEA0183, commande :

```
sudo gpsd -N -n -G /dev/ttyACM0
```

Lancement

Une fois tous ces paramètres réglés le système complet de navigation OpenCPN / Pypilot est opérationnel, il suffit alors de lancer la commande pypilot, puis lancerOpenCPN

1 Taper la commande : pypilot

Cela à pour effet de lancer et initialiser tous les paramètres nécessaires au service Pypilot.

Voir explications des différents services et paramètres pypilot dans document précédent.

Si on veut de logs pour comprendre le processus d'initialisation de Pypilot et son fonctionnement, on peut taper la commande : pypilot > pypilot.log. (Ouvrir ensuite ce fichier texte pypilot.log pour le lire)

2 Lancer OpenCPN.

OpenCPN via l'adresse locale localhost et le port TCP 20220 déjà paramétré précédemment va pouvoir lire toutes les informations venant de ses différents capteurs: Capteur inertiel, GPS, vent etc...

Options :

Pour une meilleure ergonomie et précision dans l'utilisation d'OpenCPN sur la tablette on peut faire l'acquisition d'un souris pour la tablette Android. (souris + adaptateur USB Bluetooth)

Pour une meilleure ergonomie pour la commande de pilote plutôt qu'utiliser l'écran tactile de la tablette et ses commandes +1 +10 -10 -1 du plugin pypilot on peut opter pour l'utilisation d'une télécommande avec des touches physiques +1 +10 -10 -1. *Cela fera l'objet d'un autre document.*

Je vous engage à aller sur cette page qui comporte plein d'aides très utiles sur Pypilot en français :

<https://www.navitop.fr/documentation-fra>

ou ces liens :

<https://github.com/arthur-expeditions/pypilot-workbook-fr/wiki>

https://voiliernana.fr/aide/manuel_pypilot.pdf