**Modules GPS BlueRov**

Bruschet Louis

Baillet Rémy

Sinigaglia Célia

Li Tianxu

Leroy Nathan



Table des matières

[**I- Matériel utilisé** 2](#_Toc72081211)

[**II-Guide d’utilisation** 3](#_Toc72081212)

[1) Première utilisation 3](#_Toc72081213)

[2) Utilisation d’un script python pour récupérer les données 3](#_Toc72081214)

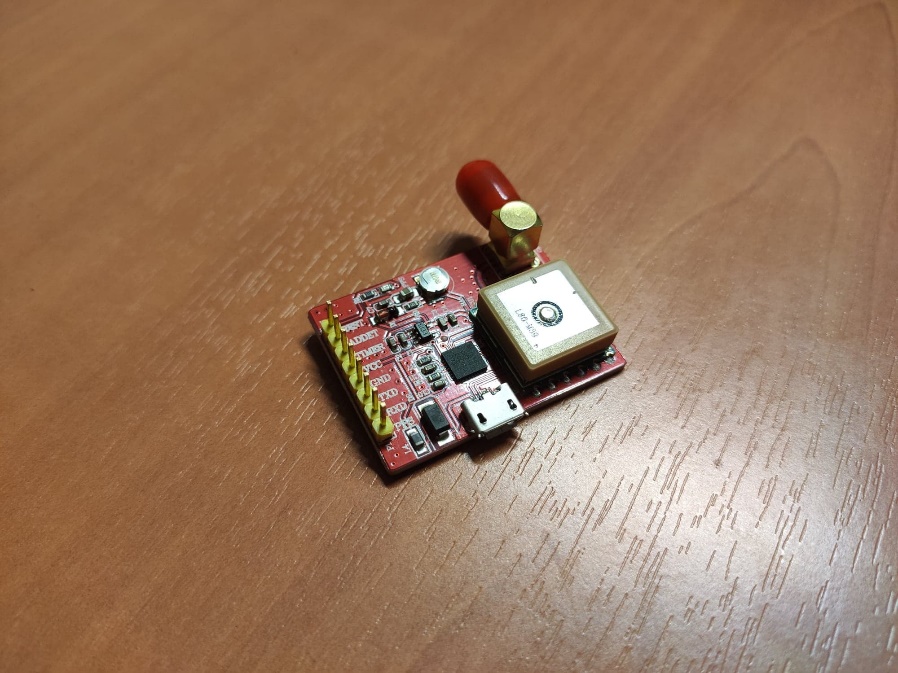
[3) Connexion avec Raspberry 4](#_Toc72081215)

[**III- Problèmes rencontrés** 4](#_Toc72081216)

[**IV- Sources** 5](#_Toc72081217)

[**V- Annexe :** 5](#_Toc72081218)

# **I- Matériel utilisé**

Pour la réalisation du module GPS du BlueRov, nous avons utilisés un Raspberry Pi 4 pour les tests. Il est nécessaire de savoir que le BlueRov utilise un Raspberry Pi 3B, modèle recommandé par le fabriquant du module GPS. Pour ce qui est de ce dernier, nous nous sommes penchés sur un module GPS basé sur un L80 pouvant être lié par liaison TTL ou par liaison USB à la Raspberry. Il est à noter qu’avec un Raspberry 4B, une mise à jour du firmware vers sa version Beta est nécessaire. Le module utilisé ressemble à cela :

Pour améliorer les performances de ce module, nous ajouté une antenne. D’après notre expérience, cela est fortement conseillé, notamment en milieu fermé où on capte très peu de satellites.

Le modèle que nous avons choisi et qui a fait ses preuves même en espace quasi-clos est le modèle GPS Ant. AT-65, ressemblant à ceci :





# **II-Guide d’utilisation**

## 1) Première utilisation

Lors de la première utilisation, il est nécessaire de réaliser les commandes suivantes sur votre terminal afin de valider le fonctionnement de votre GPS :

1 : sudo apt-get update && sudo apt-get -y install gpsd gpsd-clients python-gps

Cela va permettre d’installer les packages nécessaires pour utiliser le GPS. Pensez, si vous avez une Pi 4b, à réaliser la mise à jour du firmware vers la version Beta.

2 : sudo systemctl enable gpsd.socket

Grâce à cette commande, on active le GPS.

3 : sudo systemctl start gpsd.socket

Grâce à cette commande, on démarre le GPS.

4 : sudo systemctl restart gpsd.socket

Grâce à cette commande, on le relance le GPS.

5 : sudo systemctl status gpsd.socket

Grâce à cette commande, on vérifie que le GPS fonctionne bien.

6 : Il faut ensuite modifier le paramètre DEVICE disponible au chemin suivant : in/etc/defalt/gpsd. Puis il suffit de le remplacer par ttyUSB0 (si connecté par câble USB).

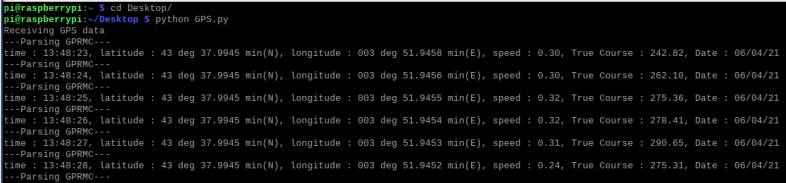
7 : Pour finir, redémarrez l’appareil (4) et utilisez la commande suivante : sudo cgps -s

Un tableau avec des données devrait apparaitre. Si vous ne captez pas assez, la date de votre GPS sera du 6 janvier 1980. Il faudra donc opter pour un espace plus ouvert.

## 2) Utilisation d’un script python pour récupérer les données

Pour cette partie du guide d’utilisation, nous allons vous expliquer comment nous avons lancé le module GPS en étant directement sur la Raspberry. Pour ce faire il est nécessaire de connecter sa Raspberry via VNC. Un tuto a été réalisé à ce sujet. Une utilisation d’un écran et de la Raspberry est aussi possible.

Une fois la connexion terminée, créez un fichier python et réécrivez le code disponible en annexe. Une fois cela réalisé, il suffit de se mettre dans le Desktop (cd Desktop) puis de lancer la commande suivante : python nomdufichier.py

Vous devriez alors obtenir les informations suivantes :

## 3) Connexion avec Raspberry

Cette partie n’a malheureusement pas pu être réalisée. Cependant des recherches ont été entamées. Il y a différents points à traiter pour réussir cette partie. Un premier consiste à trouver comment envoyer des données en passant par le mavlink. Celle-ci était en cours de recherche mais n’a pas aboutie. Une autre solution qui a été proposée consistait à réaliser un client serveur. Cette solution n’a pas été explorée et peut être réalisable pour lancer via l’ordinateur le script python du GPS. D’autres solutions sont possibles, il ne s’agit là que d’idées proposées.

# **III- Problèmes rencontrés**

Nous avons rencontré un problème majeur lors du développement de notre GPS. En effet, malgré la documentation qui était à notre disposition, nous obtenions une erreur relative à la connexion entre la Raspberry et le GPS. Plus exactement, elle ne détectait pas le module sur le port série.

Pour résoudre ce problème nous avons déjà fait la mise à jour firmware recommandée pour le Raspberry Pi 4B. Puis nous avons passé un bon bout de temps sur des manipulations inutiles avant de nous rendre compte que le problème pouvait être résolue de la manière suivante :

Dans notre Raspberry, nous avions importé les librairies Pyserial et Serial. Cependant, Serial était prioritaire, c’est-à-dire qu’il était toujours exécuté mais pas Pyserial. De ce fait le code ne pouvait pas fonctionner. Il suffit donc de désinstaller Serial et de ne garder que Pyserial afin de faire fonctionner le programme.

Un autre problème fut le GPS sans antenne. On avait essayé de récupérer des données sans antenne et cela ne marchait que très rarement. On a donc dû en installer et faire différents tests en extérieur. Sans antenne, le GPS ne fonctionne bien qu’en milieu ouvert comme le signale la documentation.

Nous n’avons pas rencontré plus de problème, même s’il fut difficile de trouver l’erreur de Serial. Si vous rencontrez d’autres problèmes, les sources vous permettrons certainement d’y remédier.

# **IV- Sources**

La plupart des informations fournies sont disponibles et ont été récupérées aux adresses suivantes :

<https://www.gotronic.fr/art-module-gps-usb-25951.htm#complte_desc>

<https://wiki.52pi.com/index.php/USB-Port-GPS_Module_SKU:EZ-0048>

# **V- Annexe :**

