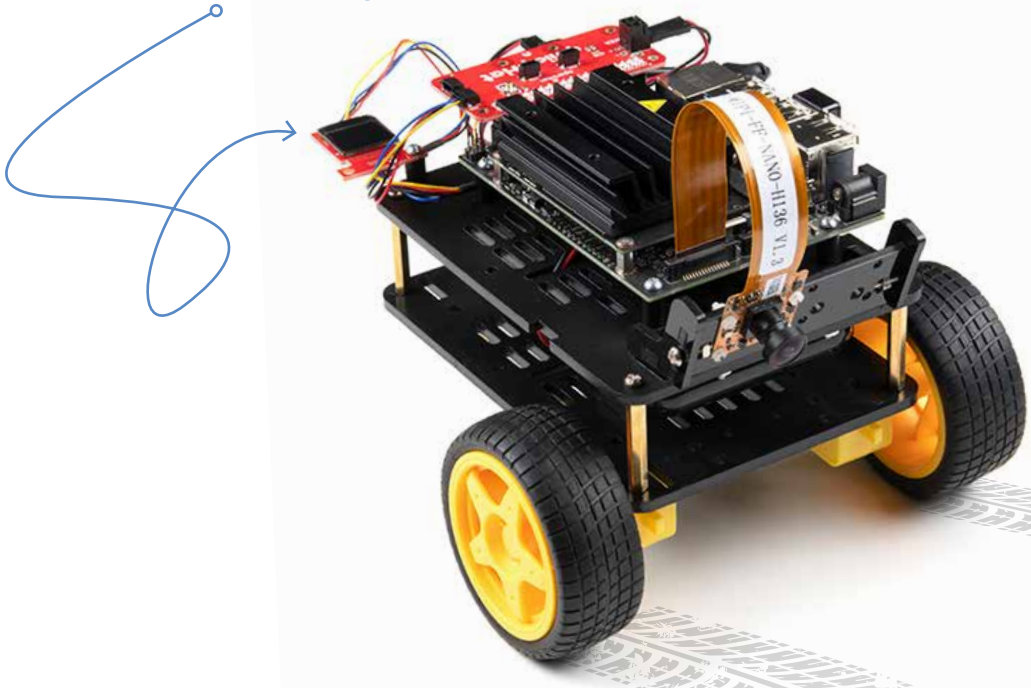
Coup de baguette magique sur le **JetBot** de SparkFun

Ou comment j'ai perfectionné mon JetBot, animé par la **carte NVIDIA Jetson Nano**



Derek Runberg (États-Unis)

Les kits électroniques sont bon marché et permettent d'aller droit au but. Surtout s'il s'agit de robots : le châssis est préassemblé, un exemple de programme et un guide détaillé sont fournis et en peu de temps votre application fonctionne. Un bon kit vous évitera un apprentissage laborieux. Vous concrétiserez rapidement votre idée initiale. L'excellence est à votre portée!

Le kit d'IA *JetBot* de *SparkFun* donne l'exemple. *Nvidia* a lancé le projet JetBot et nous y avons collaboré pour offrir à ceux qui n'ont pas d'accès à l'impression 3D la possibilité d'acquérir une version prête à monter. Ce kit JetBot comprend tout ce qu'il faut pour construire un robot expérimental, châssis et systèmes mécaniques inclus. Il comprend également les composants nécessaires pour débuter immédiatement en vision par caméra (*CV*) et apprentissage machine (*ML*) grâce à la carte *Jetson Nano*™ de NVIDIA® – rien de superflu, tout y est. Le kit d'IA JetBot de SparkFun supprime le délai d'impression 3D et la corvée du prototypage. Le gain est considérable.

On me questionne souvent sur l'extensibilité du JetBot : „est-il possible de commencer avec lui et le faire évoluer vers un *système robotique* ? La réponse est „oui !". Le JetBot est conçu pour l'expérimentation avec, hormis la caméra, des capteurs rudimentaires. Il est cependant adaptable à différents châssis, moteurs et caméras… Au lieu de me tenir à cette promesse de Gascon, je me suis retroussé les manches pour étudier de près comment procéder pour étendre les capacités de mon JetBot. Cet article décrit ma démarche et les pièges à éviter pour obtenir quelque chose de plus amusant.

Cinématique en temps réel (RTK) et à l'estime (Dead Reckoning)

Les GPS à cinématique en temps réel (RTK) reçoivent les signaux habituels des systèmes mondiaux de navigation par satellite (GNSS). Un récepteur RTK reçoit en plus un flux de messages de correction en temps réel *RTCM* (*Real Time Correction Messages*) qui lui permet d'affiner sa position avec une précision de 1 cm en temps réel. La cadence de correction varie d'un récepteur à l'autre, mais la plupart d'entre eux travaillent à 1 Hz, certains atteignent 20 Hz.

La navigation dans une ville dense, un court tunnel ou un parking peut affaiblir le signal, voire l'occulter. En l'absence de signal GNSS, la navigation à l'estime consiste à déterminer la position instantanée à partir des dernières données connues de position, de cap et de vitesse. L'estimation exploite les données fournies par la centrale inertielle 3D (IMU) et les données de déplacement du véhicule (par ex. capteurs de rotation des roues et odomètres) pour actualiser sa position.

Cinématique temps réel :
https://fr.wikipedia.org/wiki/Cinématique_temps_réel

Planification

Pour avancer avec mon JetBot, j'ai d'abord défini ce dont je disposais pour le modifier. Avec la Jetson Nano NVIDIA, il y a un impératif : pour intégrer les cartes de notre écosystème *Qwiic*, il faut que *Python* les prenne en charge, je n'ai guère envie de me coltiner des primitives I²C en Python pour l'instant.
Grâce au dépôt *GitHub* de SparkFun pour la bibliothèque *Qwiic_Py*, j'ai pu rapidement trouver les articles disponibles. À partir des différents répertoires de pilotes, j'ai compté 20 cartes intégrables. Le JetBot utilise déjà deux de ces 20 cartes : le pilote de moteur Qwiic et l'afficheur *OLED* Qwiic.
J'ai scruté la liste à la recherche des cartes les plus utiles pour le transformer en robot d'avant-garde. Après mûre réflexion, j'ai choisi trois cartes susceptibles de libérer un potentiel maximal (**fig. 1**).

1. Carte d'GPS-RTK-SMA ZED-F9P

À mon avis, un robot haut de gamme ne peut se passer d'un *GPS*, sinon comment faire en extérieur loin du bureau. Le module *ZED-F9P* de *u-blox* est récent et performant. Il exploite tous les avantages de cette technologie, notamment la cinématique en temps réel (RTK) pour obtenir une précision centimétrique si j'utilise une station de référence. Le GPS sera utilisé à la fois comme système de navigation et comme moyen de télésurveillance. En combinant la vision par ordinateur et le GPS, il serait possible de cartographier différents objets : rochers, plantes ou même personnes.

2. Carte Auto Phat de SparkFun

L'*Auto Phat* est une carte d'extension (partielle, d'où le p de *Phat*) *Raspberry Pi* compatible avec la carte NVIDIA Jetson. Cette carte et le pilote de moteur Qwiic livré avec le JetBot de série exploitent le même pilote de moteur, mais l'Auto Phat apporte d'autres fonctions et remplace avantageusement une série de cartes Qwiic. L'Auto Phat comprend le contrôleur de moteur, des entrées encodeurs de moteur, une *centrale inertielle (IMU)* et des sorties de servocommande. En remplaçant le pilote de moteur simple par l'Auto Phat, non seulement j'améliore la précision de la commande des moteurs, mais je peux piloter des servos et j'ai une IMU en prime.

3. VL53L1X Télémètre à temps de vol

Le JetBot est livré avec un capteur : une caméra. La vision par ordinateur est puissante. Ce que le JetBot peut en faire avec un peu d'apprentissage est étonnant. Avec quelques capteurs supplémentaires, le robot peut identifier des objets et mesurer à quelle distance ils se trouvent. Ou mieux encore, détecter des objets d'intérêt en dehors du champ de vision de la caméra.

Il existe de nombreuses possibilités pour la télédétection. J'ai opté pour un capteur à temps de vol (*ToF*) peu onéreux, à faible consommation avec une portée convenable pour la taille du JetBot. J'ai élu le *VL53L1X* pour sa portée de quatre mètres et son champ de vision relativement étroit. J'espérais pouvoir l'aligner avec le centre de vision du robot et obtenir une distance approximative de l'objet en vue de la caméra - plus facile à dire qu'à faire avec un objectif à très grand angle.

Intégration du matériel

J'ai commencé par l'Auto Phat, car elle nécessitait d'intervenir sur les systèmes existants et la construction de mon JetBot de série. Elle remplace le Phat Qwiic simple et le pilote de moteur Qwiic, il fallait donc les retirer du JetBot. De plus, comme l'Auto Phat a des entrées d'encodeurs, j'ai remplacé les motoréducteurs d'origine par des modèles avec encodeurs (**fig. 2**).
Ce projet était relativement simple. Le plus difficile a été d'acheminer efficacement les câbles du moteur vers la Phat. Cependant, leurs fils sont assez longs, il a donc été possible de les faire passer à travers le châssis jusqu'aux bornes à vis de la Phat, et je les ai branchés selon le guide de la Phat.
Une fois la Phat remplacée, il ne me restait plus qu'à rebrancher le câble Qwiic de l'afficheur OLED ainsi que le câble USB de la batterie (dans le port USB de la Phat), et tout était intégré !

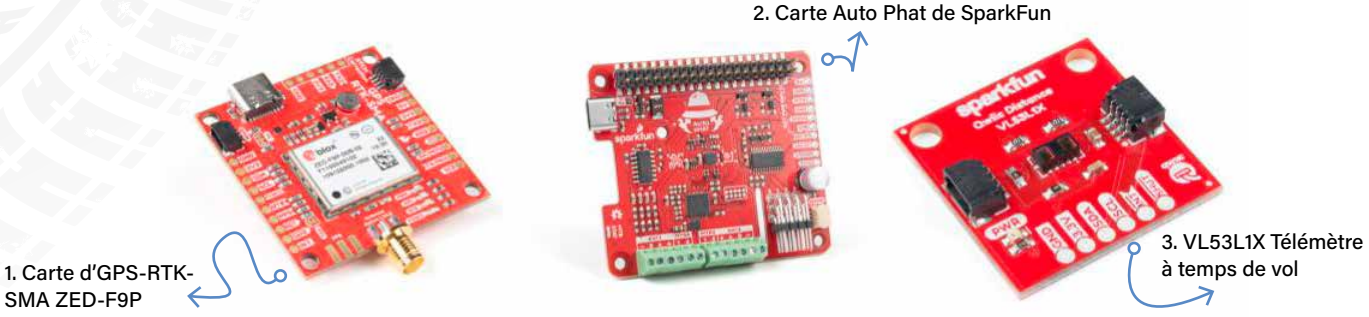


Figure 1. Les trois cartes SparkFun retenues pour perfectionner le JetBot.

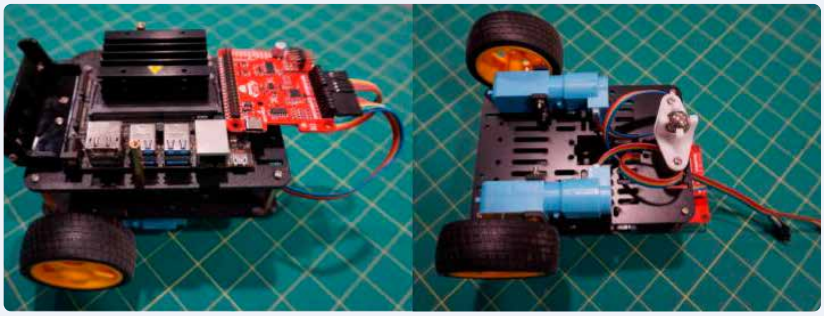


Figure 2. Sur le JetBot modifié, l'Auto Phat remplace le Qwiic Phat simple et le pilote de moteur Qwiic.

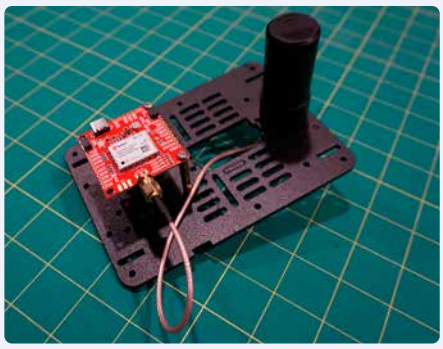


Figure 3. La carte GPS montée sur un châssis séparé, installée au-dessus de la Jetson Nano et de la monture de la caméra.

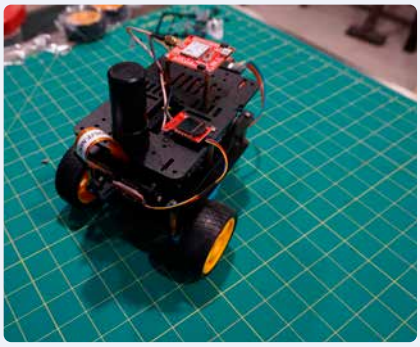


Figure 4. Vue arrière du JetBot modifié, avec plaque de montage supplémentaire pour fixer la carte GPS (sur des entretoises) et la carte OLED.

Figure 5. Télémètre ToF compatible Qwiic et module de caméra montés sur un support de fixation au JetBot.

Figure 6. Aspect final du JetBot perfectionné.

Ce fut ensuite le tour du GPS. La carte elle-même est un peu plus grande que notre carte Qwiic standard de 2,5 × 2,5 cm, il n'était donc pas possible de la monter sur l'Auto Phat. Où monter une antenne sur le JetBot déjà très encombré ? Un bout de châssis trouvé dans ma réserve de pièces a fait l'affaire, je l'ai monté au-dessus de la Jetson Nano et du support de caméra (**fig. 3**). Cela m'a ouvert un espace pour monter ensuite le GPS et mon capteur télémétrique (**fig. 4**). Le GPS s'est bien installé dans les fentes, et pour monter l'extension SMA j'ai foré un trou de taille convenable dans le châssis. Comme elle communique par liaison UART, la carte GPS fonctionne à merveille sur un ordinateur monocarte. Je pouvais la brancher à un des UART du connecteur, mais j'ai préféré utiliser le port USB, déjà configuré en UART d'origine. Inutile de chercher midi à 14 h. Le dernier ajout prévu au JetBot était le capteur de distance à temps de vol. Par bonheur, ce capteur a l'encombrement standard Qwiic : 2,5 × 2,5 cm. Je l'ai monté dans le même plan, à côté de la caméra en utilisant un support simple de mon cru et un carré de *bande Velcro* (**fig. 5**). Le Velcro offre une certaine souplesse de montage du capteur sans perçage. Il ne me restait plus qu'à trouver la longueur convenable de câble Qwiic pour relier le capteur à la carte Qwiic la plus proche de la chaîne. Avec ma configuration, c'était la carte d'affichage OLED sur le châssis ajouté. La **fig. 6** montre le résultat.

Intégration du logiciel

Il y a plusieurs façons de gérer l'intégration du logiciel à ce robot, mais la plus simple est d'utiliser l'image personnalisée pour le SparkFun JetBot. Celle-ci comprend le pilote de moteur Qwiic de SparkFun, ainsi que des exemples choisis par NVIDIA pour le JetBot et qui s'appuient sur *Jupyter Notebooks* pour accéder au système de fichiers et exécuter des scripts Python à distance.

À partir de cela, vous pouvez alors installer le paquet Python *Qwiic_py* sur votre JetBot et commencer à tester le programme d'exemple pour chaque carte. En ligne de commande sur le terminal et avec `pip` c'est rapide. Ouvrez une fenêtre de terminal sur votre NVIDIA Jetson Nano depuis le bureau ou Jupyter Notebooks, et tapez :

```
sudo pip install sparkfun-qwiic
```

Cette commande télécharge et installe les pilotes pour toutes les cartes Qwiic actuellement prises en charge par l'écosystème. Pour finir, il faut installer *PySerial*, la méthode de base de communication par *USB* entre la Jetson Nano et la carte GPS F9P. Pour installer le paquet PySerial, tapez :

```
sudo pip install py_serial
```

Les deux ensembles logiciels installés, nous allons exécuter un programme d'exemple pour chaque capteur ajouté. Le dépôt Github Qwiic_py fournit des exemples de programmes Python

pour chacune de nos cartes Qwiic prises en charge [1]. Téléchargez ou copiez-collez les exemples dans un fichier Python et exécutez-les depuis la ligne de commande en tapant :

```
python3 exemple.py
```

Si vous exécutez un script en rapport avec le GPS connecté via USB à votre Jetson Nano, vous devrez utiliser `sudo` dans votre commande, car cela lui confère la permission de lire et écrire des messages UART via USB :

```
sudo python3 exampleGPS.py
```

Note : si vous travaillez à partir de Jupyter Notebooks et que vous voulez utiliser le GPS ou exécuter un script nécessitant une connexion UART, vous devrez modifier les permissions de votre port UART. Vous pouvez le faire à l'aide de la commande suivante :

```
sudo chmod 660 /dev/ttyAMC0
```

Si vous êtes curieux, j'ai déposé sur GitHub quelques exemples de programmes Python créés en utilisant le JetBot modifié via Jupyter Notebook. Vous pouvez les trouver et les télécharger sur votre propre JetBot [2]. Chaque Jupyter Notebook fournit une explication des extraits de code et permet de les exécuter directement, sans ligne de commande. Si vous ne connaissez pas encore Jupyter Notebooks, consultez le tutoriel rapide [3].

Conclusion

Nous n'avons fait qu'effleurer le sujet de la mise à niveau de votre JetBot. Il y a une infinité de capteurs et d'actionneurs faciles à intégrer et à utiliser avec les ordinateurs monocartes. Les capacités d'apprentissage automatique et d'intelligence artificielle de la Jetson Nano de NVIDIA en font un outil de premier plan pour conférer des performances hors norme aux plateformes robotiques les plus simples.

Moyennant quelques accessoires, j'ai mis à niveau le JetBot lui-même, mais pourquoi s'arrêter là ! Si germe en vous l'idée de commander avec la Jetson Nano une plateforme robotique de votre choix, osez, le monde bientôt vous appartiendra !
L'an passé, j'ai porté JetBot sur la plateforme robotique *RVR* de *Sphero* avec quelques composants du catalogue SparkFun. Un peu d'usinage, ponçage et perçage et le RVR devient plus intelligent et tire parti du *Machine Learning* de la Jetson Nano. Vous pouvez trouver mon tutoriel étape par étape ici [4].

La Jetson Nano de NVIDIA et la plateforme JetBot sont parmi les kits et produits robotiques les plus excitants que j'ai utilisés depuis longtemps. Ils marient admirablement technologie de pointe, accessibilité et souplesse. Si vous voulez les bidouiller, ne vous gênez pas. Je ne me lasse pas de perfectionner ce robot, tout en développant mes compétences en apprentissage automatique et en intelligence artificielle.

(200689 – VF : Yves Georges)

SparkFun & NVIDIA

L'apprentissage automatique (*Machine Learning* ou ML) est nouveau pour nous tous ! La coopération entre SparkFun et Nvidia a débouché sur des innovations concrètes accessibles à nos clients. Nous proposons depuis plusieurs années des robots doués d'auto-apprentissage sous forme de kits autour de la Jetson Nano et de tutoriels en ligne gratuits. NVIDIA repousse constamment les limites du ML tandis que SparkFun s'active à simplifier ces techniques.

LIENS

[1] Exemple de programme Python pour les cartes Qwiic : https://bit.ly/2KNPf5k
[2] Exemples de programmes Python dans Jupyter Notebook : https://bit.ly/3a9RbO7
[3] Tutoriel de Jupyter Notebook : https://bit.ly/36eanJA
[4] Tutoriel sur l'univers du RVR de Sphero : https://bit.ly/3c7Non3

Accessoires

Vous trouverez chez Elektor et Sparkfun les accessoires mentionnés dans cet article.

- Kit JetBot AI Kit v2.1 de SparkFun basé sur une Jetson Nano
  www.elektormagazine.fr/esfe-en-jetbot1
- Carte GPS-RTK-SMA de SparkFun - ZED-F9P (Qwiic)
  www.elektormagazine.fr/esfe-en-jetbot2
- SparkFun Auto pHAT pour Raspberry Pi
  www.elektormagazine.fr/esfe-en-jetbot3
- Carte télémètre SparkFun - 4 m, VL53L1X (Qwiic)
  www.elektormagazine.fr/esfe-en-jetbot4