Een upgrade voor uw SparkFun JetBot

hoe ik de functionaliteit van mijn JetBot (met de Jetson Nano van NVIDIA) heb uitgebreid

Derek Runberg (USA)

Elektronicakits moeten in eerste instantie zuinig zijn en zo eenvoudig mogelijk. Bij robotica betekent dat een kant-en-klaar chassis, dito voorbeeldcode en zelfs een handleiding om zo snel mogelijk iets aan de praat te krijgen. Een goede kit maakt dat mogelijk: geen steile leercurve en vooral een snel resultaat. Wij noemen dat "Snel van Nul naar Geweldig".

De SparkFun JetBot AI-kit is een voorbeeld van die aanpak. Onze versie van het open source-project JetBot, dat is opgezet door NVIDIA, was een samenwerking met NVIDIA om klanten die geen toegang hebben tot een 3D-printer de mogelijkheid te bieden om een voorbereide versie te kopen. Deze versie van de JetBot levert het materiaal om snel een tafelrobot te bouwen, inclusief het chassis en alle benodigde mechanische systemen, zoals op de kopfoto te zien is. De kit bevat ook de benodigde componenten om basale camera vision (CV) en machine learning (ML) zonder extra componenten te implementeren met behulp van de NVIDIA® Jetson Nano™: geen franje, alleen wat noodzakelijk is. Met de SparkFun JetBot AI-kithoeft men niet op een 3D-printer te wachten en niet met vroege prototypes te klungelen.

Vaak vragen mensen naar de schaalbaarheid van de JetBot: "Kan iemand hiermee beginnen en het dan opschalen tot een robotsysteem?" Het antwoord is: "Ja!" Het JetBot-systeem is ontworpen als een tafelrobot met relatief kleine, eenvoudige sensoren (afgezien van de camera). Maar het kan gemakkelijk worden aangepast en

opgeschaald naar verschillende chassis, motoren en camera's. Dat bleef voor mij tot nog toe theorie, dus ik besloot de mouwen op te stropen, er in te duiken en te kijken welke onderdelen zinvol waren om mijn JetBot op te leuken. Dit artikel beschrijft hoe ik dat deed en welke valkuilen ik tegenkwam bij het gebruik van de JetBot-kit als basis voor iets dat... ~~geavanceerder~~ leuker was.

Planning

Met mijn JetBot in de hand moest ik als volgende stap bij het aanpassen van de JetBot bepalen wat er beschikbaar was. Als ik bij de Jetson Nano van NVIDIA boards uit ons Qwiic-ecosysteem wilde gebruiken, moesten deze voor eenvoudige integratie in Python ondersteund worden (ik heb geen zin om I²C zelf in Python te programmeren).
Ik ging naar SparkFun's Github-repository voor de Qwiic_Py-bibliotheek en daar kon ik snel zien welke boards ik zou kunnen gebruiken. Ik vond in de diverse driver-directories twintig verschillende board-opties die ik kon integreren. Van die twintig gebruikt de JetBot er al twee: de Qwiic Motor Driver en het Qwiic OLED Display. Toen ik de lijst nader keek vroeg ik me af: "Wat is zinvol om aan mijn robot toe te voegen om hem naar een hoger niveau te tillen?" Na enig nadenken kwam ik uit op een drietal boards die de meeste mogelijkheden zouden bieden (**figuur 1**).

1. GPS-RTK-SMA ZED-F9P Breakout

Bij high-end robotica denk ik meteen aan GPS. Zou het niet mooi zijn om robotica buitenshuis te doen en niet thuis op tafel? De ZED-F9P-module van u-blox bevat de nieuwste en mooiste technologie met alle toeters en bellen, inclusief de mogelijkheid om Real Time Kinematics (RTK) te doen om een nauwkeurigheid op millimeterniveau te halen bij gebruik van een basisstation. Voor mijn geestesoog zag de GPS al gebruikt worden voor navigatie en om op afstand cartografie te bedrijven. De combinatie van computervision en GPS zou het in kaart brengen van allerlei objecten, zoals rotsen, planten of zelfs mensen, mogelijk maken.

2. SparkFun Auto pHAT

De SparkFun Auto pHAT is een Raspberry Pi pHAT (*partial hat*) die ook compatibel is met de NVIDIA Jetson. Hij heeft eigenlijk dezelfde motordriver-chip als de Qwiic Motor Driver die deel uitmaakt van de oorspronkelijke kit, maar voegt daar meer functionaliteit aan toe in de vorm van een pHAT in plaats van een reeks Qwiic-boards.

Real-Time Kinematica (RTK) en gegist bestek

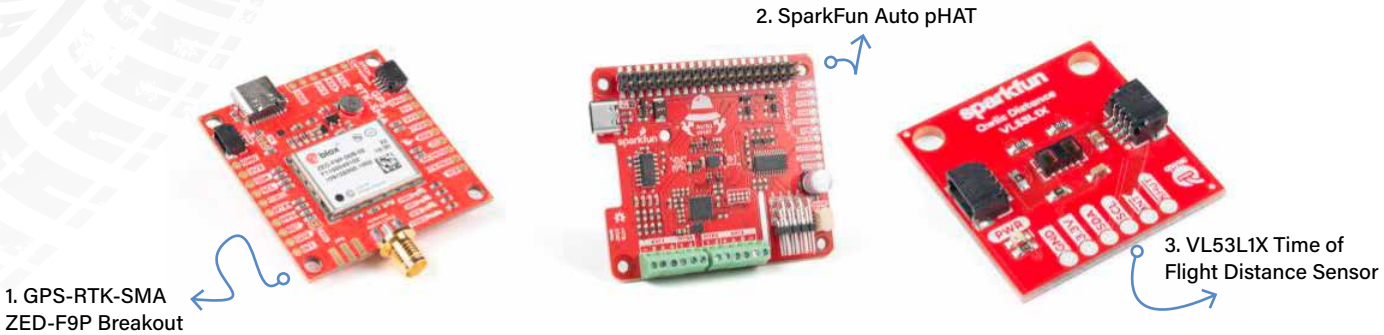
GPS-ontvangers die geschikt zijn voor RTK ontvangen de normale signalen van de wereldwijde satellietnavigatiesystemen (GNSS), plus een correctiesignaal om een positienauwkeurigheid van 1 cm te bereiken. Bovenop deze signalen ontvangt een RTK-ontvanger een RTCM-correctiestroom en berekent dan zijn locatie met een nauwkeurigheid van 1 cm in real time. De snelheid varieert van ontvanger tot ontvanger, maar de meeste zullen tenminste eenmaal per seconde een update leveren. Sommige ontvangers halen zelfs 20 keer per seconde.
Navigatie door een dichtbevolkte stad, een korte tunnel of een parkeergarage kan een slechte signaalkwaliteit of volledig signaalverlies opleveren. Dat soort problemen zijn op te lossen met 'gegist bestek': daarbij wordt huidige positie bepaald door eerdere positiegegevens te combineren met de momentele snelheid en koers. 3D-versnellingssensoren (IMU's) en afstandsgegevens van het voertuig (bijvoorbeeld wielomwentelingen en snelheidsmeters) kunnen worden gebruikt om de huidige positie van een voertuig voortdurend te berekenen wanneer GNSS-gegevens tijdelijk uitvallen.

De Auto pHAT bevat de motordriver, ingangen voor motor-encoders, een IMU en uitgangen voor servomotoren. Door de eenvoudige motorbesturing te vervangen door deze pHAT, kan ik niet alleen de motoren nauwkeuriger besturen, maar ook servo's en de IMU gebruiken.

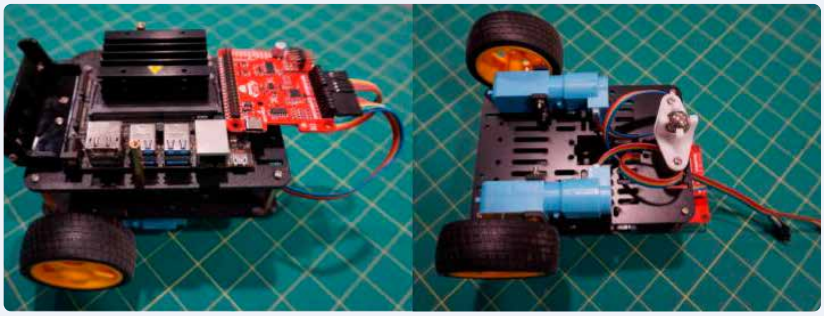
3. VL53L1X Time of Flight Distance Sensor

De JetBot wordt geleverd met één sensor: een camera. Computer vision is erg krachtig, en het is verbazingwekkend wat de JetBot kan doen met alleen computer vision en wat ML. Maar door een paar extra sensoren toe te voegen kan de robot objecten identificeren en de afstand tot die objecten bepalen. Of, nog beter, interessante objecten detecteren buiten het gezichtsveld van de camera.

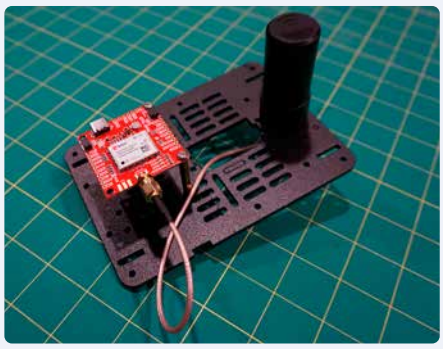
Er zijn veel mogelijkheden voor afstandsmeting. Uiteindelijk heb ik gekozen voor een Time of Flight (ToF) sensor die goedkoop en zuinig is en een bereik heeft dat past bij de schaal van de JetBot. De keuze viel op de VL53L1X-afstandssensor vanwege het bereik van vier meter en het relatief smalle gezichtsveld. Ik hoopte dat ik de sensor zou kunnen uitlijnen met het middelpunt van het gezichtsveld van de robot en een ruwe afstand zou kunnen bepalen tot het object dat in het zicht van de camera was; gemakkelijker gezegd dan gedaan met een fish eye-lens.



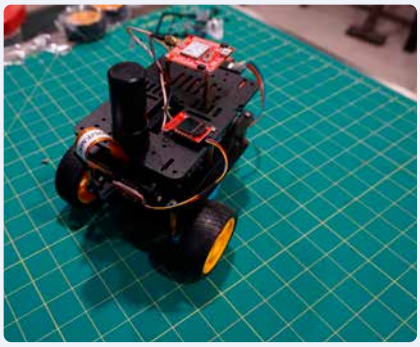
Figuur 1. De drie SparkFun-boards die ik heb gekozen om de JetBot uit te breiden.



Figuur 2. Op de gemodificeerde JetBot vervangt de Auto pHAT de eenvoudige Qwiic pHAT en de Qwiic-motordriver.



Figuur 3. De GPS-print is op een apart subchassis gemonteerd zodat die boven de Jetson Nano en de camerabeugel past.



Figuur 4. Achteraanzicht van de aangepaste JetBot met de extra montageplaat om de GPS-print (op afstandsbussen) en de OLED-print vast te zetten.

Figuur 5. De Qwiic-compatibele ToF-afstandssensor en de cameramodule zijn met een beugel op de JetBot gemonteerd.

Figuur 6. De constructie van de opgevoerde JetBot.

Hardware-integratie

Ik begon met die systemen te integreren die de bestaande systemen van mijn standaard JetBot zouden beïnvloeden: de Auto pHAT. Die vervangt de eenvoudige Qwiic pHAT en de Qwiic Motor Driver, dus die moesten allemaal worden verwijderd. Omdat de Auto pHAT ingangen heeft voor motor-encoders, heb ik bovendien de standaard motoren met tandwielvertraging vervangen door motoren met encoders (**figuur 2**).

Dit onderdeel van het project was relatief eenvoudig. Het moeilijkste was nog om een manier te vinden om de motoraansluitingen efficiënt naar de pHAT te leiden. De draden zijn gelukkig vrij lang, dus het was geen probleem om ze door het chassis naar de schroefklemmen op de pHAT te voeren, en ik bevestigde ze conform de *Hookup Guide* voor de pHAT.

Nadat de pHAT was vervangen, hoefde ik alleen maar de Qwiic-kabel voor het Qwiic-OLED-display en de USB-kabel van het accupack in de USB-poort van de pHAT te steken, en alles was geïntegreerd!

De volgende stap was de GPS-unit. De print zelf is iets groter dan onze standaard Qwiic-printafmetingen van 2,54 × 2,54 cm), dus montage op de Auto pHAT zat er niet in. Om het nog ingewikkelder te maken, moest ik ook nog een plaats vinden voor de antenne op de toch al overvolle JetBot. Omdat ik in ieder geval wat meer ruimte nodig had, heb ik een stuk chassis uit mijn knutselkist gebruikt en dat boven de Jetson Nano en de camerabevestiging gemonteerd (**figuur 3**). Daardoor had ik meer ruimte om later zowel de GPS als mijn afstandssensor te monteren (**figuur 4**). De GPS paste netjes in de gleuven, en ik boorde een gat in het chassis om de SMA-aansluiting te monteren.

Het GPS-board werkt het best samen met enkelkaartcomputers via UART. Hoewel ik de kabel had kunnen aansluiten op een van de UART's op de header, koos ik ervoor om de USB-poort als UART-poort te gebruiken omdat deze standaard al zo is geconfigureerd, wat weer een hoop gedoe scheelde.

De laatste toevoeging voor de JetBot was de ToF-afstandssensor. Gelukkig heeft deze sensor onze standaard Qwiic-vormfactor van 1 inch bij 1 inch. Ik heb hem recht boven de camera gemonteerd met behulp van een eenvoudige zelfgemaakte beugel en een vierkant stukje klittenband (**figuur 5**). Ik heb hier klittenband gebruikt omdat ik daardoor de sensor flexibel kan monteren zonder een heleboel gaten te boren. Als laatste, na het monteren van de sensor, moest ik een Qwiic-kabel van de juiste lengte vinden en deze aansluiten op het dichtstbijzijnde Qwiic-board in de keten. In mijn configuratie was dat het OLED Display-board op het extra chassis. Het resultaat is te zien in **figuur 6**.

Software-integratie

Er zijn natuurlijk verschillende manieren om de software-integratie met de JetBot aan te pakken, maar het is het eenvoudigste om het aangepaste image voor de SparkFun JetBot te gebruiken. Dit integreert SparkFun's Qwiic-motordriver met de JetBot-pakketten, en ook met de voorbeelden die NVIDIA gebruikt voor het koppelen van de JetBot met Jupyter Notebooks om toegang te krijgen tot het bestandssysteem en Python-scripts op afstand uit te voeren. Daar kunt u dan op doorborduren door het *Qwiic_py* Python pakket op uw JetBot te installeren en de voorbeeldcode voor elk board

SparkFun en NVIDIA

Machine Learning (ML) is nieuw voor ons allemaal! Om het voor onze klanten begrijpelijk te houden, terwijl we zelf voortdurend innoveren, hebben we een community nodig. SparkFun werkt al een aantal jaar met NVIDIA samen om machine learning naar onze klanten te brengen in de vorm van kits rond de Jetson Nano en van gratis online lesmateriaal. NVIDIA verlegt constant de grenzen van ML en SparkFun helpt om die technologie begrijpelijk te maken.

te testen. Dat gaat snel vanaf de opdrachtregel en `pip`. Open een terminalvenster op de NVIDIA Jetson Nano vanaf de desktop of Jupyter notebooks, en tik het volgende commando in:

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

Dit commando downloadt de stuurprogramma's en installeert al onze momenteel ondersteunde Qwiic-boards. De laatste installatie die moet worden gedaan met `pip` is het installeren van het PySerial-pakket, de primaire communicatiemethode via USB tussen de Jetson Nano en het F9P GPS-board. Geef het volgende commando in om het PySerial-pakket te installeren:

```
sudo pip install py_serial
```

Als beide pakketten zijn geïnstalleerd, kunnen we voorbeeldcode draaien voor elk van de extra sensoren. Voorbeelden van Python-code voor elk van onze ondersteunde Qwiic-boards in de Qwiic_py GitHub-repository zijn te vinden op [1]. Download of copy&paste de voorbeeldcode in een Python-bestand en voer dat uit vanaf de opdrachtregel door het volgende commando in te typen:

```
python3 example.py
```

Als u een script uitvoert dat werkt met de GPS-ontvanger die via USB op uw Jetson Nano is aangesloten, moet u `sudo` gebruiken, omdat u rechten nodig hebt om te lezen van en te schrijven naar de UART-verbinding over USB:

```
sudo python3 exampleGPS.py
```

Let op: als u vanuit Jupyter Notebooks werkt en de GPS wilt gebruiken (of een ander script dat toegang tot een UART-verbinding nodig heeft), dan moet u de toegangsrechten van uw UART-poort wijzigen. U kunt dat doen met het volgende commando:

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

Als u nieuwsgierig bent naar een paar voorbeeldprogramma's in Python die ik heb gemaakt met de aangepaste JetBot in Jupyter Notebook-vorm, kunt u die vinden op GitHub via [2] en downloaden naar uw eigen JetBot. Elk Jupyter Notebook geeft uitleg over de code-snippets en geeft de mogelijkheid om de code van daaruit te draaien, zonder dat u de commandoregel nodig hebt. Als Jupyter Notebooks nieuw voor u is, bekijk dan de korte handleiding op [3].

Tot slot

We hebben alleen een tipje van de sluier opgelicht over de mogelijkheden om uw JetBot op te voeren. Er zijn heel veel verschillende sensoren en actuatoren die gemakkelijk met enkelkaarcomputers kunnen worden geïntegreerd en gebruikt. De ML- en AI-mogelijkheden van de NVIDIA Jetson Nano maken hem een koploper voor high-end prestaties, zelfs op de eenvoudigste robot-platforms.
Met behulp van een paar accessoires heb ik de JetBot zelf opgevoerd, maar dat is nog niet het einde van het verhaal! Als u eenmaal gaat nadenken over de Jetson Nano als besturingssysteem voor een robotica-platform, gaat er een wereld voor u open en kunt u kijken wat voor uitbreidingen u nog meer kunt toevoegen en of hij misschien ook met andere platforms te gebruiken is.
Vorig jaar heb ik het JetBot-platform met een paar onderdelen uit de SparkFun catalogus overgezet op het Sphero RVR-robotplatform. Met wat snijden, schuren en boren kon ik de RVR aanpassen om de ML-mogelijkheden van de Jetson Nano te gebruiken. Mijn stap-voor-stap-handleiding is te vinden op [4].
De NVIDIA Jetson Nano en het JetBot-platform zijn misschien wel de leukste roboticakits en -producten die ik in lange tijd heb gebruikt. Ze combineren geavanceerde technologie met laagdrempeligheid en flexibiliteit. En er is ruimte voor eigen aanpassingen. Ik ben constant op zoek naar manieren om de robot te verbeteren en tegelijkertijd mijn kennis en begrip van ML en AI te vergroten.

200689-04

WEBLINKS

[1] Voorbeeld Python-code voor Qwiic-boards: https://bit.ly/39CB5O2
[2] Voorbeeld Python-programma's in Jupyter Notebook: https://bit.ly/3a9RbO7
[3] Jupyter Notebook-tutorial: https://bit.ly/36eanJA
[4] Tutorial voor het aanpassen van de Sphero RVR: https://bit.ly/3c7Non3

Gerelateerde producten

Bent u op zoek naar de belangrijkste producten die in dit artikel genoemd zijn? Elektor en SparkFun staan voor u klaar!

- SparkFun JetBot AI Kit v2.1 Powered by Jetson Nano
  www.elektormagazine.nl/esfe-en-jetbot1
- SparkFun GPS-RTK-SMA Breakout - ZED-F9P (Qwiic)
  www.elektormagazine.nl/esfe-en-jetbot2
- SparkFun Auto pHAT for Raspberry Pi
  www.elektormagazine.nl/esfe-en-jetbot3
- SparkFun Distance Sensor Breakout - 4 Meter, VL53L1X (Qwiic)
  www.elektormagazine.nl/esfe-en-jetbot4