Communicatie met Pi en Lidar

# Introductie

Best is om hier de grafische interface te gebruiken van de Pi om de nodige files te bekijken.

$ startx

Het communiceren gebeurt als volgt :

* Bij het opstarten van de Rasberry Pi (Pi) word de RoboPeakLidar (Lidar) afgezet, door deze via python te openen (kleine truc om ervoor te zorgen dan hem niet constant draait.
* Een startcode wordt dan via matlab gestuurd, 201, naar de Pi, op poort 7070 (cfr. initLidar)
* Op de Pi start dan de python script die input 201 op poort 7070 verwacht.
  + Deze bevind zich in
* Deze start de Lidar (het begint te draaien), doordat de C++ file wordt uitgevoerd
* De C++ file start en zal continu data doorsturen, naar de PC, op poort 7070

# How to cross compile

Volg de stappen beschreven in : <http://www.hertaville.com/development-environment-raspberry-pi-cross-compiler.html>

Het kan aangenamer zijn om op de grafische interface van de Pi te zijn (statx in terminal) en deze website op de Pi te lezen, zodat copy past van command line mogelijk is (Pi moet natuurlijk op internet aangesloten zijn ! Dit kan door direct een lan kabel te gebruiken, of door in de terminal $ wicd-curses te typen (de wifi manager die wie momenteel op de Pi hebben)

## Practische zaken voor eclipse

Nadat jullie de tutorial hebben doornomen (het is geen slecht idee om een helloworld project aan te maken en zien of deze kan runnen op de Pi !), moeten jullie dus het RPLidar C++ project kopiëren en er een andere van maken, bvb RPLiar\_Pi. Deze moet je weer linken naar de libreries/sources zoals hieronder beschreven :

1. Install from Eclipse
2. Unzip C++ code
3. Create a C++ project in Eclipse
4. Drag the unzipped files in the project
5. Add the necessary paths in eclipse.
   * right click project
   * C/C++ build
   * Settings
     + GCC C++
       - Include, add the following paths (click on workspace, add graphically)
         * sdk/include
         * sdk/src
         * skd/src/arch
         * sdk/src/hal
6. Add the necessary libraries :
   * right click project
   * C/CC++ build
   * Settings
     + GCC C++
       - Linker libraries
         * pthread
7. Build
8. Left click on arrow next to run
   * Run configurations
   * new launch configurations
   * Arguments → /dev/ttyUSB0 (for example)

Oppassen, in op de Pi heet de lidar /dev/rplidar of /dev/lidar (check door ls /dev/ te gebruiken !).

Men kan ook simpel wer, via de terminal naar de workspace, project, debug of release folder gaan en als input argument /dev/lidar geven :

$ cd to//workspace/project/release/  
$ ./nameofyourproject /dev/lidar

./ staat voor execute !

Ik weet dat er nog een error kwam, bij het compileren van de cross compiler (een bepaalde time functie die niet in die bepaalde cross compiler was). Dit heb ik (denk ik) opgelost door een extra librery toe te voegen naast pthread.

**Als jullie errors hebben onder linux, niet bang zijn om te google !**

Linux heeft een zeer grote community die op een efficiënte manier fouten kan oplossen door gebruik te maken van de terminal : Zoals jullie hebben gezien, door gewooon een error de copy pasten (dat die couldn’t bind to /dev/ttyUSB0) kan met een copy past to terminal oplossing vinden !

## Alternatief bij het kopiëren van C++ project

Ik heb niet de remote acces gevold volgens de website, ik heb een makkelijkere oplossing toegepast :

Door het commando secure copy te gebruiken kan ik de hele folder van mijn gecompileerde project op de laptop, naar de Pi doorsturen, indien ze op hetzelfde netwerk zijn (bvb wifi roomba, echte verbinding met internet is dus NIET nodig !)

$ scp -r path/to/your/directory/you/want/to/copy username@ipadress:/path/to/the/dir/you/want/to/put/your/folderss