README Raspberry Pi

1 Setup Raspberry Pi 2 or 3

- Download the latest version of Raspbian (as of writing Stretch) from https://www.raspberrypi.org/downloads/raspbian/ .
- Write the Raspbian img-file to an SD card. Do **not** copy the file to the SD-card, it wont work. Check https://www.raspberrypi.org/documentation/installation/installing-images/ if you are unsure.
  - First, make sure that you have extracted the img file from the downloaded zip.
  - If you are using Windows/OS X or are a Linux GUI fan, download an image burner software, like Etcher, and use that program to write the image to the SD-card.
  - If you like Linux terminal solutions:
    - Make sure to unmount (umount) the SD-card partitions, probably located down the /media directory.
    - Run "dd" from the command line and then flush the disks by running "sync":
      - $ sudo dd if=~/Downloads/2017-08-16-raspbian-stretch.img of=/dev/mmc<something> bs=4M
      - $ sync
- Connect the keyboard, mouse and monitor to the raspberry.
- Put the SD-card into the raspberry and boot.
- Connect to WiFi.
- Start a terminal window and type "sudo raspi-config"
  - In Localisation Options, select the appropriate timezone.
  - In Interfacing Options, enable SPI and I2C and the SSH interfaces.
- Type command "sudo apt-get update" to fetch info about the latest software version.
- Type command "sudo apt-get dist-upgrade" to update to the latest software version.
- Reboot the Raspberry Pi.
- Connect once again to the Raspberry and find the wireless IP address:
  - Type "ifconfig wlan0" and SSH to this address in the future.

2 Development environment

The software can be built either on a standalone Linux system or directly on the Raspberry Pi. Both methods should work equally well.

2.1 Setup for development on Raspberry Pi

Make sure that the following packages are installed: gcc, make. Use "apt-get install [package]" if needed.

2.2 Setup for development on standalone Linux system

The instructions are verified for Debian-based Linux distributions (such as Ubuntu).

Make sure that the following packages are installed: gcc-arm-linux-gnueabihf, make Use "apt-get install [package]" if needed.

3 Distributed files

Extract the zip-file you got from Acconeer and look at the file structure.

- makefile and rule/ contain all makefiles to build the example programs.
- lib/*.a are pre-built Acconeer software.
- include/*.h are interface descriptions used by applications.
- source/example_*.c are applications to use the Acconeer API to communicate with the sensor.
- source/acc_board_*.c are board support files to handle target hardware differences.
- source/acc_driver_*.c are hardware drivers supposed to be customized to match target hardware.
- source/acc_os_*.c is the operating system support module. It is not meant to be modified, but is provided for reference.
- doc/ contains HTML documentation for all source files. Open doc/html/index.html .
- out/ contains pre-built applications (same as executing "make" again).

4 Building the software

- Enter the directory that you extracted in section 3.
- To build the example programs, type "make" (the ZIP file already contains pre-built versions of them).
- All files created during build are stored in the out/ directory.
- "make clean" will delete the out/ directory.

5 Executing the software

First you need to transfer the executable to the Raspberry Pi (unless the zip-file was already extracted to the Raspberry Pi.).

Then start the application using: - ./out/example_detector_distance_and_service_rpi_<board and sensor version>