Getting Started with Crazyflie for SSY191

March 2019

This document is intended as a short guide for getting acquainted with the workspace in the course Model-Based Development of Cyber-Physcial Systems (SSY191).

1 Get the workspace to your computer

1.1 Prerequisites

You have several different alternatives for the development of software for the Crazyflie (including getting a Virtual Machine or getting the Windows installer from the Bitcraze website). In this document, we support using the Virtual Machine from Bitcraze. You are still free to use another development way (for example the Windows installer), but since we have more experience using the Virtual Machine, we might not be able to help as much if you do not use the VM.

First, download the latest version (2018.01) of the Bitcraze Virtual machine (`https://wiki.bitcraze.io/projects:virtualmachine:index`). Import the VM into Oracle Virtualbox (`https://www.virtualbox.org/`). We have developed the environment in MATLAB 2016b – this means that 2016b is ideal to use. You **cannot** use an older version than 2016b. Using a newer version will work. but if you have different versions in your group, you can get compatibility problems (the ones with the newest MATLAB must export their changed Simulink files to older versions of MATLAB).

NOTE! The icon "Update All Projects" on the desktop of the Bitcraze Virtual Machine should **not** be pressed under any circumstance (no matter what it says on the Bitcraze webpage). If you have already pressed it, you have to delete your current VirtualMachine and reload it from scratch.

1.2 Possible error

Some of you might get the following error when launching BitcrazeVM on VirtualBox: end Kernel panic - not syncing: Attempted to kill init!. To fix this you need to increase available RAM for the virtual machine (e.g. from 1024 MB to 4096 MB), however this might unfortunately result in the following error: VT-x is disabled in the BIOS for all CPU modes (VERR_VMX_MSR_ALL_VMX_DISABLED).

To fix **this** error, have a look at this video explaining how to enable VT-x: `https://www.youtube.com/watch?v=7U-e3Ui-NhU`.

Required MATLAB apps If you are unable to generate code from Simulink it might be due to that your MATLAB installation does not have all the required apps installed. To install additional apps open MATLAB and click on the tab "APPS" and then "Get More Apps". This will open the "Add-On Explorer" in which you can search for apps for MATLAB and install them. The ones that are required are:

- "`Simulink`" by MathWorks
- "`Simscape`" by MathWorks
- "`MATLAB coder`" by MathWorks
- "`Simulink coder`" by MathWorks
- "`Embedded coder`" by MathWorks

1.3 Sharing folder to the VM

To have the correct workflow, you need to share the crazyflie folder with the VirtualMachine, so that any changes you make to files in the folder in your host operating system (e.g. Windows) can be accessed in the VirtualMachine. Create an **empty** folder on your host operating systems (name it for example `Bitcraze_Shared_Folder`). This is most easily done by turning off the virtual machine and opening the shared folder settings. Share the `Bitcraze_Shared_Folder` folder and make sure that you **check** the **auto-mount setting**.

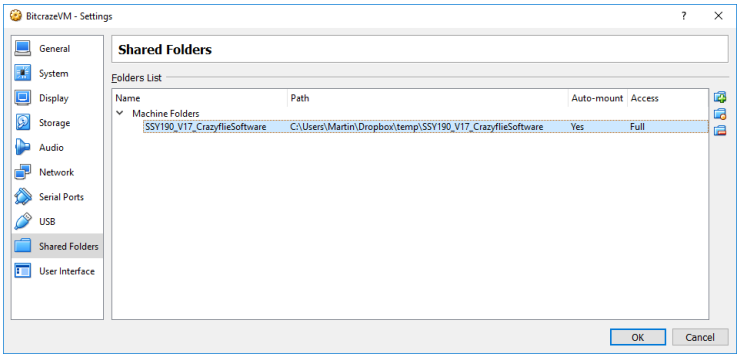

Figure 1: Adding a shared folder to the virtual machine.

After starting the virtual machine again, the folder will be mounted automatically and can be seen from the file manager. However, you will not be able to open it. Open a terminal and type the following command to allow the bitcraze user access to virtualbox shared folders:

`sudo adduser bitcraze vboxsf`

The password for the bitcraze user is `crazyflie`.

When this is done, you need to either reboot the virtual machine or log out and log in again.

1.4 Installing QtSvg package

Inside the VM, open a terminal and type:
`sudo apt-get install python3-pyqt5.qtsvg`

If you need to type the password, it is (as before) `crazyflie`. When it asks if you want to continue to use more disk space, type `y`.

1.5 Git classroom - creating your repository

Each Pingpong group will have their own Git repository on Github. Git is a tool for sharing the code you write together with your classmate, and it is very useful for example when you are several people editing the same code at once. For a quick git tutorial, see `https://try.github.io/levels/1/challenges/1`. If you are wondering which commands you actually will use, it's mostly `git add`, `git commit -m "Your special commit message"`, `git push` and `git pull`. This is summarized in Figure 2.

In this course we use Github classroom. To get started, visit `https://classroom.github.com/g/hrZvnoRx`. Each Pingpong group should have a team with their group number as the name, for example "Group 15" if you happen to be in group 15 on Pingpong. After your group has been created, you will get your own git repository, which will be found at
`https://github.com/SSY191-master/quadrotor-lab-spring-2019-Group-XX`, where XX is your group number. At this address, you can click the green button **Clone or download** and copy the adress you will use to clone this git repo.

Now open the Virtual Machine. To be able to copy commands, go into **Devices**, then under **Shared Clipboard** select **Bidirectional**. Open the shared folder (e.g. `Bitcraze_Shared_Folder`), then right-click and press "Open Terminal". Copy the following command into the terminal:

```
git clone --recursive https://github.com/SSY191-master/quadrotor-lab-spring-2019-Group--XX.git
```

Here, you must replace the url with *the url found from pressing the button "Clone or download" earlier*!

1.6 Generating C code and uploading to quadrotor

Create a suitable controller by changing the Simulink file "crazyflie.slx" in your **host OS** (e.g. Windows). Press "Build" and wait for the code generation to finish. Now go into the Bitcraze VM, enter the directory called *crazyflie-firmware*, right-click and select "Open terminal here", then type `make`.

Now make sure that the Bitcraze Bluetooth dongle is connected via USB, and also in the VM right-click the USB-icon in the bottom right and select Bitcraze Crazyradio PA Dongle. Start the quadrotor in bootload mode (hold

Figure 2: How git will likely be used in this course (mostly). Credit: XKCD.

the start button for a few seconds until the blue lights start blinking) and then type `make cload` in the terminal.

1.7 Changing channel of your quadrotor

To make it easier to connect to the correct quadrotor (and not some other groups), you should change the channel you send on to **your Pingpong group number**. This is done in these steps:

1. Connect to your Crazyflie using the BitCraze Client
2. Select "Connect", then "Configure 2.0"
3. Change the channel to your **PingPong group number**.

Unfortunately, this does not help when running "make cload".

2 Repository contents

The cloned repository consists of two folders at the top level:
`crazyflie-clients-python` and `crazyflie-firmware`.

`crazyflie-clients-python` contains python clients from Bitcraze which are needed to load new firmware onto the quadrotor. You will not need to touch anything in this folder.

The other folder, `crazyflie-firmware`, contains the source code for the firmware. In addition to the original source code, the folder `simulink-model` contains the templates for the Simulink and Simscape models that you will use during the course. The contents of this folder is as follows:

- closed_loop_script.m - Script for running tests with Breach. Will be used in later exercises.
- closed_loop.slx - Closed loop system. You are not expected to do any changes in this file, in fact it is easier for us to help you if you keep the structure as is.
- crazyflie.slx - Controller and code generator. Your controller goes in this file.
- crazyflie_wrapper.tlc - Code generation template. Inserts the generated code at the right place in the firmware with the correct inputs and outputs. Does not need to be changed.
- open_loop.slx - Open loop simulink file. Should not need any changes.
- params.m - Set up parameters, such as mass and inertia.
- plant.ssc - Simscape model of the plant. Your model goes here.

- sensors.sscp - Simscape blackbox model of the sensor emulator. Can not be changed.

NOTE! As you can see, the majority of your work will be performed in the Simulink file crazyflie.slx and the Simscape file plant.ssc.

3 Overview of the closed-loop Simulink model

An overview of the closed-loop Simulink model provided in the SSY190 repository can be found in Fig. 3. It consists of three different parts which are detailed more carefully in the coming sections of this document. **NOTE!** You do not need to change anything in this model! This section is only for getting confident with the different parts of the closed-loop model used in the course.

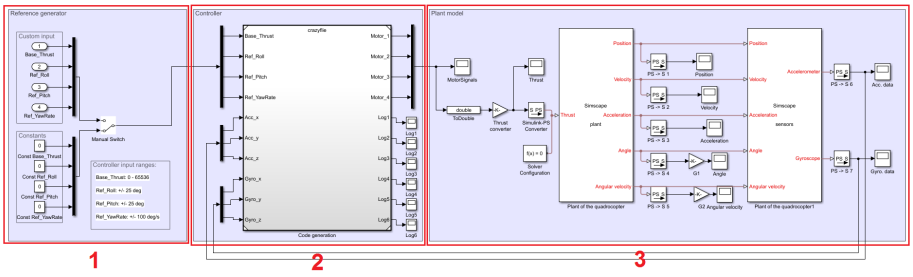

Figure 3: An overview of the Simulink system used in the SSY190 quadrotor project.

3.1 Reference generator

Fig. 4 shows the reference generator, which is the first component in the Simulink system. The inputs you are able to give to the system are

- The base thrust to the motors of the quadcopter. This is a number in the interval $[0, 65536]$, where 0 corresponds to 0% motor power and 65536 corresponds to 100% motor power.
- The reference signal for the roll angle, given in degrees.
- The reference signal for the pitch angle, given in degrees.
- The reference signal for the yaw angle rate, given in degrees per second.

There is a manual switch which you can *double-click* to switch between getting inputs from the MATLAB workspace (input blocks), and generating the inputs by setting constant values to the reference signals.

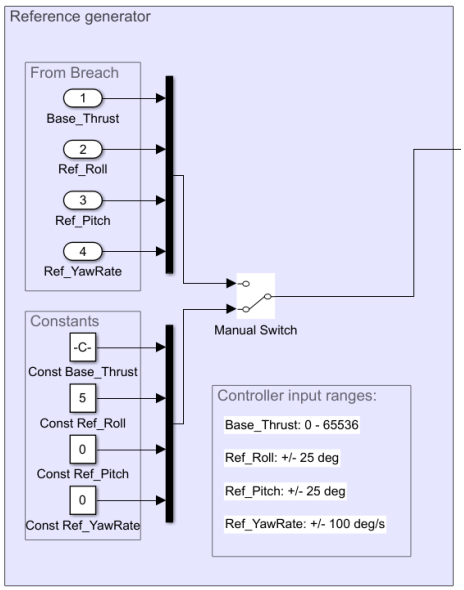

Figure 4: The first part of the closed-loop system, the reference generator.

3.2 Controller

Fig. 5 shows the controller block, which is the second component in the Simulink system. Double-clicking this block will open the Simulink model `crazyflie.slx`, which is the file where you have to implement your controller for the quadcopter. Fig. 6 shows an empty controller block. Clicking the build button within this file will generate new c-code for the firmware. **NOTE!** If you get an error, you will need to change the compilation toolchain (the standard one set in `crazyflie.slx` is configured for Linux). You do this by entering the Configuration Parameters, selecting "Code Generation" in the list to the left, and then choosing the toolchain "MinGW64 v4.x | gmake (64-bit Windows)" in the drop-down list labeled "Toolchain".

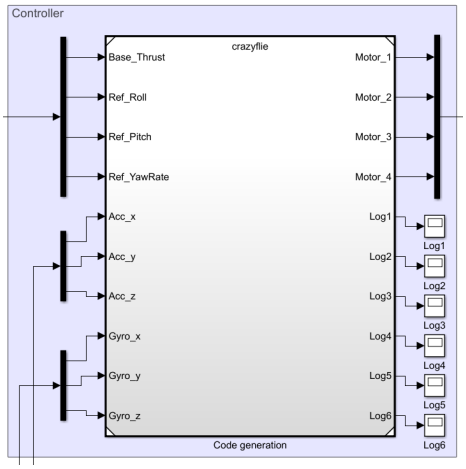

Figure 5: The second part of the closed-loop system, the controller block.

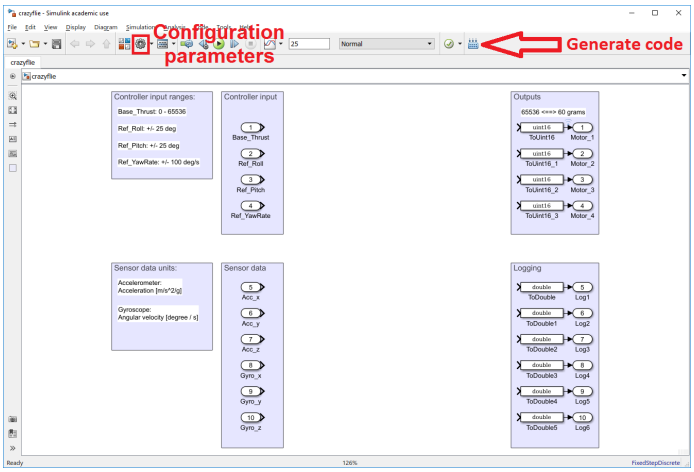

Figure 6: Clicking "build" in the controller block will generate new code. **NOTE!** If this does not work, you have to change the compilation toolchain in the Configuration Parameters. Click the Configuration Parameters, select "Code Generation", and then choose another toolchain in the drop-down list labeled "Toolchain". For 64-bit Windows, the recommended toolchain is "MinGW64 v4.x | gmake (64-bit Windows)"

3.3 Plant model

Fig. 7 shows the plant model, which is the third component in the Simulink system. Double-clicking the block called "Simscape plant" will open the `plant.ssc` file, which is where you will model the plant using Simscape. There are also several different scopes available, which you can use to check that your outputs from the model make sense when you are developing the Simscape code.

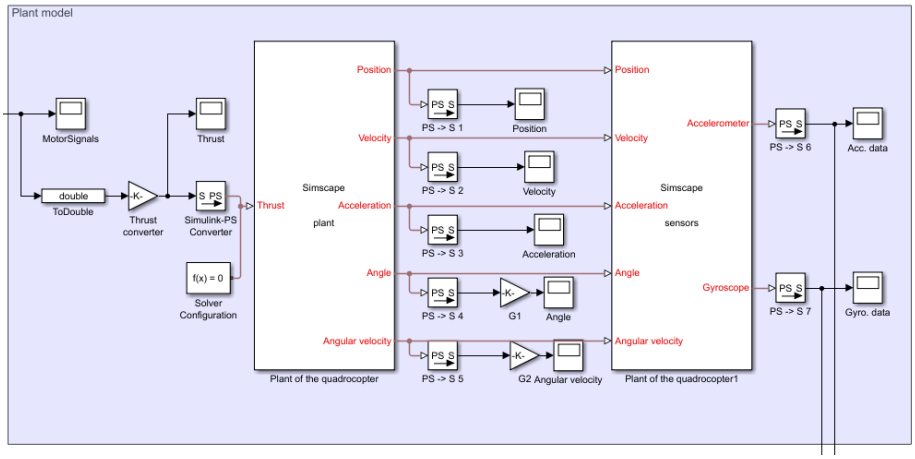

Figure 7: The third part of the closed-loop system, the plant model.