Applications of ML in Mechatronics Tools GUI Technical Manual

This guide is prepared as a reference for you to know the necessary setup needed to operate the GUI. Additionally, it is meant to serve as a brief tutorial to help you understand some of the basics of the Raspberry Pi, esepecially if you are using it for a project.

1. Setup the Raspberry Pi

1.1. Prepare the Raspberry Pi (RPi) SD Card

The Raspberry Pi used in this course is a Raspberry Pi Zero W, where W stands for Wireless, which is the new version that has built-in WiFi and Bluetooth. Although WiFi is not strictly needed for the purpose of this application, it is used in this manual to make the setup easier and improve the overall useability.

1.1.1. Write the image to the SD Card

To run an operating system (OS) on the raspberry pi, you will need to burn it onto the SD card. Follow the following steps to burn the OS image:

1. Download and install the **Raspberry Pi Imager** from here
2. Run the imager
3. For the operating system, choose **Raspberry PI OS (Other)** and then **Raspberry Pi OS Lite**
4. For the SD Card, insert the micro-SD card into your computer and select it
5. Click **Write**, and Confirm if prompted
6. When the image is written, you may need to remove and re-insert the SD card to be able to continue

Note that the SD Card root folder will be referred to as `boot` or `BOOT` folder in the rest of this manual

1.1.2. Enable Wireless Networking

To enable wireless networking, you will need the RPi to connect to your local wireless network. This will allow you to access the RPi from your computer through Secure Shell protocol (SSH).

You will need to define a `wpa_supplicant.conf` file for your particular wireless network. Put this file in the `boot` folder, and when the Pi first boots, it will copy that file into the correct location in the Linux root file system and use those settings to start up wireless networking. After the Pi is connected to power, make sure to wait a few (up to 5) minutes for it to boot up and register on the network. The Pi's IP address will not be visible immediately after power on, so this step is crucial to connect to it headlessly. Depending on the OS and editor you are creating this on, the file could have incorrect newlines or the wrong file extension so make sure you use an editor that accounts for this. Linux expects the line feed (LF) newline character. For more information, see this Wikipedia article.

`wpa_supplicant.conf` file example:

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=CA

network={
 ssid="<Name of your wireless LAN>"
 psk="<Password for your wireless LAN>"
}
```

Note that some older wireless dongles don't support 5GHz networks. Make sure to change the `<placeholders>` with your actual WiFi SSDI/name and password. Keep the quotation marks only and remove the greater and less signs. This information is case-sensitive

More information on the `wpa_supplicant.conf` file can be found here.

1.1.3. Enable SSH Access

To enable ssh, you will need to create an empty file in the **`boot`** folder called **`ssh`** . The file name should NOT have any extensions (e.g. ssh.txt is not acceptable)

1.1.4. Enable Serial Gadget Driver

As RPi Zero has one micro USB port. It is by default used to connect devices like a keyboard or a mouse through USB OTG. In our application, we want to use this usb port instead for serial communication with the computer. Accordingly, some settings need to be changed.

1. **USB Driver Setting**

Enable **`dwc2`** driver in **`BOOT/config.txt`** , add the line at the end. At the same time you can enable/disable anything extra you might need like SPI or I2C.

```
dtoverlay=dwc2
```

2. **Modify default cmdline.txt to enable serial gadget driver**

We need to enable loading of the serial USB gadget driver.

In **`BOOT/cmdline.txt`** :

Add `modules-load=dwc2,g_serial` to the kernel command line file, `cmdline.txt` in `boot`. Note the `root=PARTUUID=xxxxxxxx-yy` part, don't modify that. You may need to power cycle it one more time after the first boot for the USB configuration to work.

After adding the new snippet, the content of the file should look like this:

```
console=serial0,115200 console=tty1 root=PARTUUID=e8af6eb2-02 rootfstype=ext4 elevator=deadline fsck.repair=yes rootwait quiet init=
```

1.2. Connect to your Board

First of all, insert the prepared micro SD card to the RPi. Use the USB cable to connect the Raspberry Pi to your computer. Make sure to connect to the port labelled **USB** NOT the one labelled **PWR**. After a few minutes, the Raspberry Pi should be running and you will need to confirm two things:

1. The Raspberry Pi is showing as a COM port. You can confirm that through the Device Manager in Windows.
2. The Raspberry Pi has connected successfully to your local wireless network and has obtained an IP Address. At this point, you should also note down the IP Address of the RPi. To find the IP address, there are multiple ways. You can check the **Retrieving your Raspberry Pi's IP Address from another device** section here as an example. One method that I find particularly easy is to try to SSH to the RPi directly using `ssh pi@raspberrypi` and enter the default password `raspberry` and then use `hostname -I` when connected to get the IP Address. You can also just use the **Get RPi IP** option from the **Utilities** menu in the GUI.

1.3. Setup a method to transfer files to the Raspberry Pi

You will ned a way to transfer files from your computer to the Raspberry Pi. This will be mainly needed in order to copy the trained models files to the Raspberry Pi. There are multiple ways to do that such as copying files over SSH or setting up a samba server. However, we will go over only one method here that works for Windows.

1. Download WinSCP from here
2. Install your favorite layout. I chose **Explorer**
3. When it opens setup the following in **Session**:

```
File Protocol: SCP
Host name: The IP address of you Raspberry Pi
User name: pi
Password: raspberry
```

4. **(This is Optional)** In **Advanced > SCP /Shell**:

```
Shell: sudo su - (if you want to be able to write to protected directories)
```

5. Connect and you can transfer files just by dragging and dropping.

1.4. Clone the python script

1. SSH to the raspberry pi using `ssh pi@<IP_ADDRESS>` and then enter the password, which is `raspberry` by default
2. Create a folder in the Raspberry Pi, let's create it in the home directory. First `cd ~` and then `mkdir SFU_ML` to create the directory
3. open the folder by `cd SFU_ML`
4. Install Git to the RPi using `sudo apt-get install git`
5. Initialize a repo through `git init`
6. Allow cloning subdirectories using `git config core.sparsecheckout true`
7. To choose the right folder to clone, use `echo 'RPi_Script*' >> .git/info/sparse-checkout`
8. Add the remote repo using `git remote add -f origin https://github.com/RamyE/SFU_ML.git`
9. Clone the folder using `git pull origin master`
10. Set master to track the remote master branch using `git branch --set-upstream-to=origin/master master`
11. Later when you want to get the latest updates from the repo, you can go to the folder and just use `git pull`.

1.5. Download Python 3 and the required dependencies

1. SSH to the Raspberry Pi using `ssh pi@<ip_address>` in a command prompt and then enter the password
2. Install python3.7 using `sudo apt-get install python3.7`
3. Install pip using `sudo apt-get install python3-pip`
4. Install other dependencies using `sudo apt-get install libatlas-base-dev`
5. Install the required dependencies using `pip3 install -r /home/pi/SFU_ML/RPi_Script/requirements.txt`

1.6. Setup the python script to run automatically on startup

1. SSH to the Raspberry Pi
2. Edit rc.local file using `sudo nano /etc/rc.local`
3. Add commands to execute the python program. Be sure to leave the line `exit 0` at the end. Add the following line before `exit 0`

```
su pi -c 'python3 /home/pi/SFU_ML/RPi_Script/main.py' &
```

4. Save the file and exit. In nano, to exit, type Ctrl-x, then Y, and then press Enter.

2. Set up and Run the GUI

2.1. Setup the Virtual Environment

This section covers installing a python interpreter and package installations in a virtual environment, rather than system-wide. We will use Anaconda here, however, if you have an alternative that you are using, you may continue to use that. To setup the virtual environment and install the required dependencies, follow these steps:

1. Head over to anaconda.com and install the latest version of Anaconda. Make sure to download the “Python 3.7 Version” for the appropriate architecture. Check this page for help, if needed.
2. Open the command prompt and confirm that conda is in PATH by typing `conda -V`
3. Create a new virtual environment using `conda create -n ml_course python=3.7 pip` where `ml_course` is the name of the virtual environment and could be activated later using `conda activate ml_course`
4. The required dependencies will be installed in the next section as they require the repo to be cloned first
5. Make sure not to install any other applications, especially GUI applications that use Qt, in the same virtual environment (e.g. Spyder) as they may cause conflicts. If you want to use other applications, you can install them in your **base** environment which could be activated using `conda activate base`

2.2. Clone the repo and install the dependencies

1. Clone the repo to your computer as it has the labs and GUI code. Go to the directory where you want to clone the repo and use `git clone https://github.com/RamyE/SFU_ML.git` (This will not work if you do not have git installed. You can download Git from here)
2. Navigate to the directory `Ui_project` inside the repo folder throught the command line interface, you can use `cd Ui_project`
3. Activate the right python virtual environment using `conda activate ml_course`
4. Install all the dependencies using `pip install -r requirements.txt`

2.3. Run! (but don't run away)

Now all you need to do is run the GUI. You can follow these steps:

1. Open a command line prompt
2. Navigate to the repo folder using `cd` which stands for **change directory**
3. Activate the virtual environment (don't forget this step!) using `conda activate ml_course`
4. run the GUI Application using `python .\Ui_project\main.py`

3. How to use the GUI

The GUI helps you use test your trained model on the end use platform, which is the Raspberry Pi. Before you use the GUI, you will need to have the following ready:

1. A Raspberry Pi Zero W connected to your computer.
2. A trained model generated from the lab tutorials and transferred to the Raspberry Pi using WinSCP.
3. Test Dataset without the labels in a CSV file, which could be generated as part of the lab tutorials.
4. Run the Raspberry Pi Script on the Raspberry Pi by SSH'ing to the Raspberry Pi and starting the script (this step is only needed if you did not setup the script to run automatically on startup).
5. Choose the right options in the GUI, connect to the Raspberry Pi Serial Port and then press **Start Processing**. For help with the options, you can use the **"?"** button on the top right corner of the GUI Window.