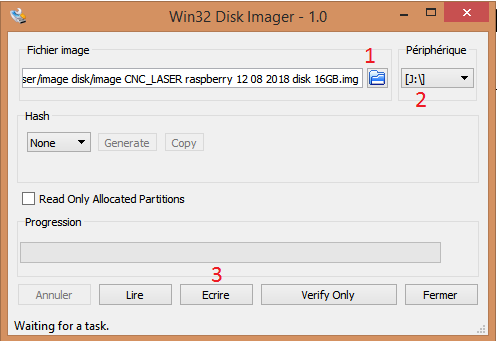
Installer Rasbian et cie pour la mini CNC Laser

English version at the midle of this paper

# Installer l’image disque Rasbian stretch lite, par copie de l’image disque sur la carte SD.

* Ouvrir Windiskimager
* Sélectionner l’image à copier sur la SD card (1)
* 
* Sélectionner le périphérique à graver (2)
* Cliquer sur Ecrire (3)

Copier les fichiers de paramétrage SSH sur la carte SD :

1. Copier/créer un fichier vide se nommant : « ssh » sans extension
2. Copier/créer un fichier se nommant « wpa\_supplicant.conf » contenant :

ctrl\_interface=DIR=/var/run/wpa\_supplicant GROUP=netdev

network={

ssid="*nom de votre box adsl*"

psk="code d’accès de votre boxe adsl, du type xx56…….R458…"

key\_mgmt=WPA-PSK

}

Mettre la carte SD dans le raspberry pi (RPI)

Brancher un écran sur le port HDMI.

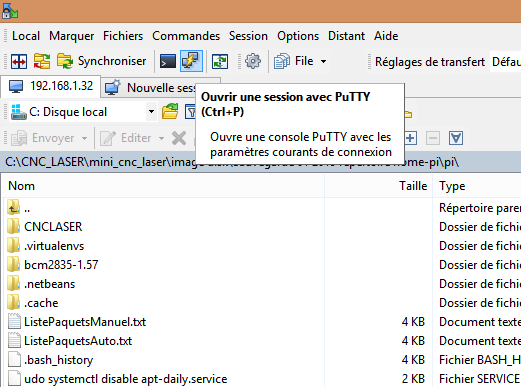
Allumer le raspberry pi.

Assurer vous d’être proche d’un accès internet par wifi.

Regarder ce qui défile à l’écran et repérer : « My IP address is 192.168.1.xx »

Noter cette adresse pour s’y connecter en SSH.

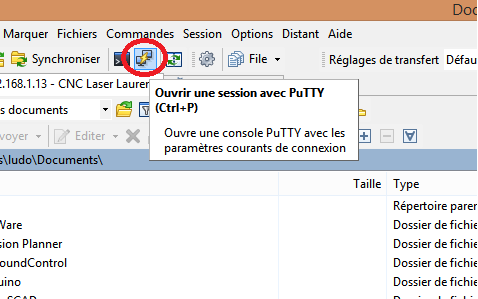
Se connecter au RPI en ssh via WinSCP (voir le tutoriel associé)



Le login est « pi »

Le mot de passe est « raspberry »

Ouvrir une session Putty :



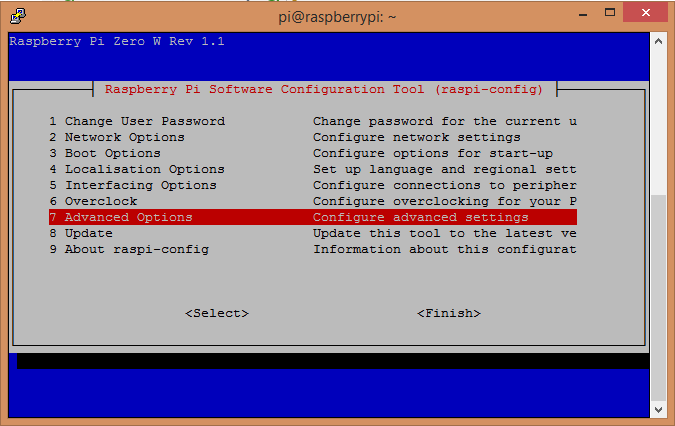
Si votre installation de Raspbian est toute neuve, vous aurez besoin d’étendre la partition afin d’utiliser tout l’espace libre. Pour cela :

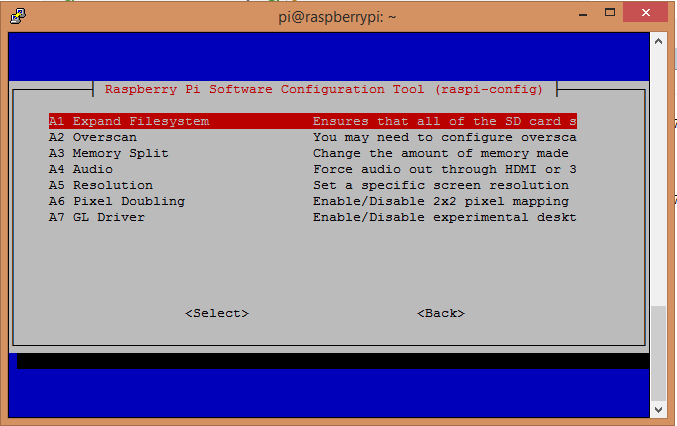
$ sudo raspi-config



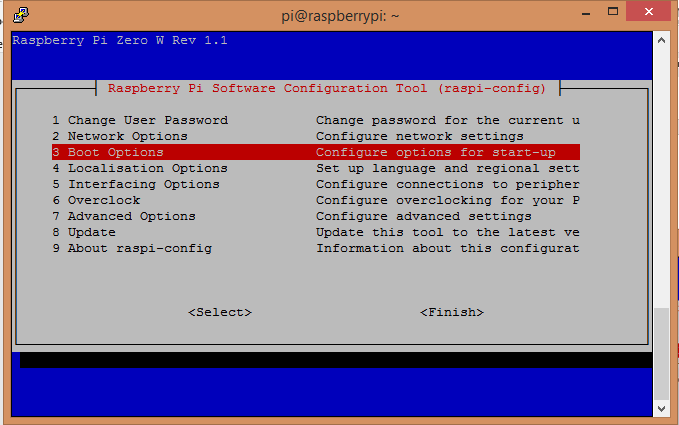
Raspi-config: Expand Filesystem

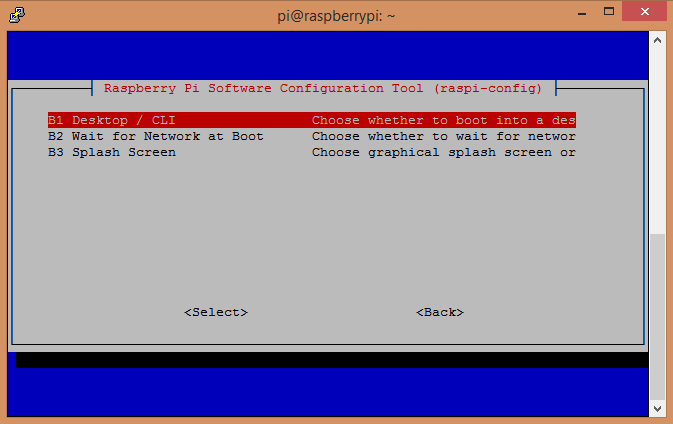
Dans le menu, allez sur l’option “**Expand Filesystem**” et appuyez sur entrer et validez.

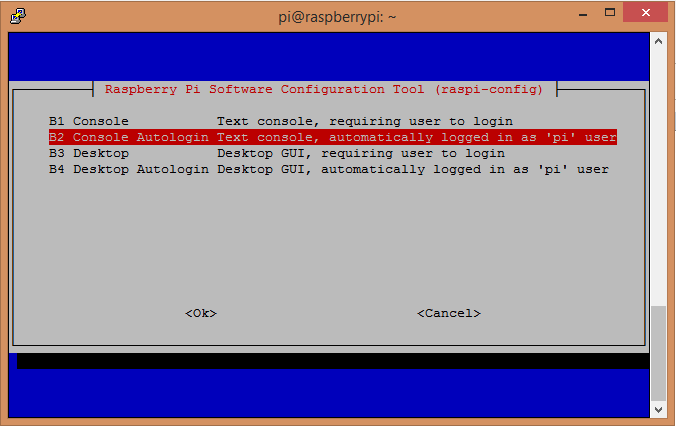




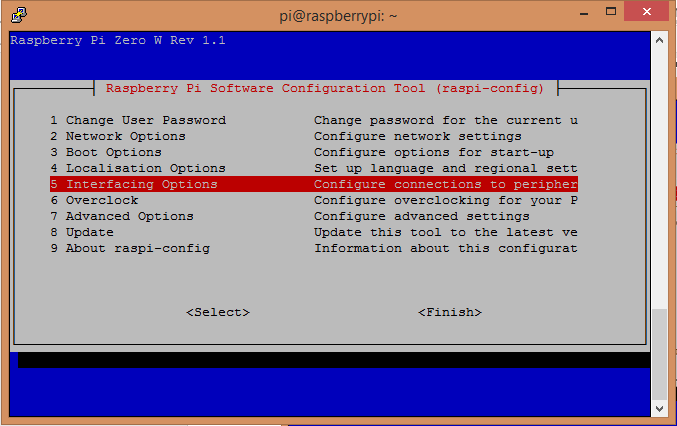
Dans Boot options, configurer le raspberry pi pour démarrer sans login.







Configuration des interfaces :



Puis cliquer sur SPI interface, et choisissez ok.

Enfin redémarrez votre Raspberry:

$ sudo reboot

Eventuellement créer mot de passe super utilisateur :

$ sudo passwd root

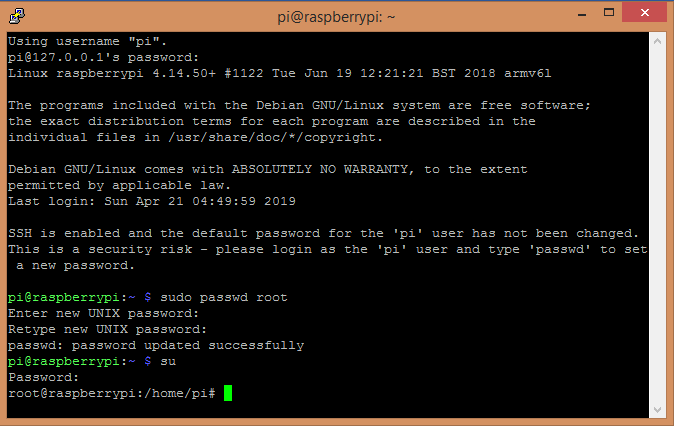
Entrer le mot de passe : « rpi »

Pour acceder en root, taper

$ su

Puis le mot de passe

La ligne de commande doit être comme ceci desormais :



Ouvir une console Putty pour installer les packages suivants :

Mise à jour des paquets :

$ sudo apt-get update

$ sudo apt upgrade

# **Installer l’environnement virtuel et python 3.5 :**

Il faut que python3.5 soit installé :

$ sudo apt-get install python3.5

Seul 3 applications doivent être globales : python3.5, pip et virtualenv. Soit :

$ sudo apt-get install virtualenv

On installe aussi PIP et PIP3 :

$ curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py

$ python get-pip.py

$ apt-get -y install python3-pip

$ apt-get update

Et une mise à jour :

$ sudo pip install --upgrade virtualenv

**creating a folder for your projects/environments**

$ mkdir CNCLASER

$ cd CNCLASER

**creating environment**

<https://virtualenvwrapper.readthedocs.io/en/latest/command_ref.html>

$ sudo pip**3** install virtualenv virtualenvwrapper

$ sudo rm -rf ~/.cache/pip

Taper les commandes suivantes :

$ echo -e "\n# virtualenv and virtualenvwrapper" >> ~/.profile

$ echo "export WORKON\_HOME=$HOME/.virtualenvs" >> ~/.profile

$ echo "export VIRTUALENVWRAPPER\_PYTHON=/usr/bin/python3.5" >> ~/.profile

$ echo "source /usr/local/bin/virtualenvwrapper.sh" >> ~/.profile

Fermer le terminal et le reouvrir

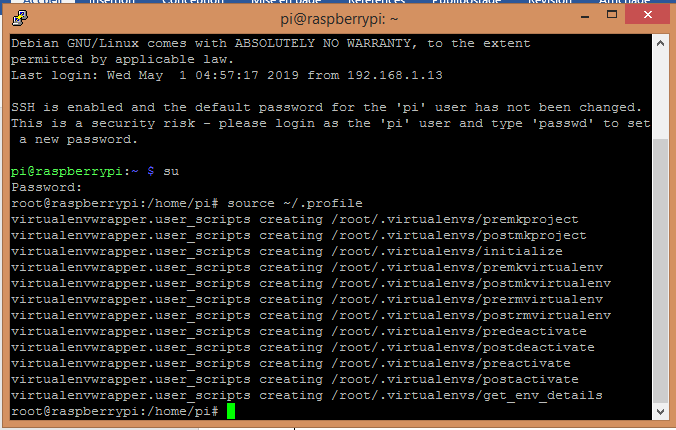
A la réouverture passer en superutilisateur :

$ su

Rentrer le mot de passe « rpi »

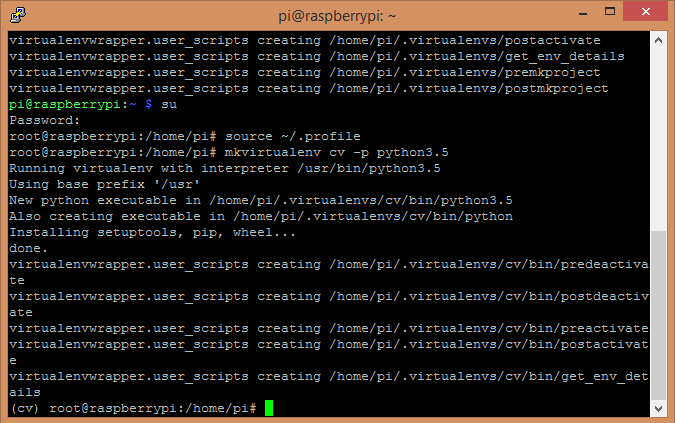
Puis  :

$ source ~/.profile



$ mkvirtualenv cv -p python3.5

Un certain temps est nécessaire à l’installation des outils de l’environnement.



Puis à chaque fois que des modifications devront être faites de l’installation raspberry pi

$ source ~/.profile

$ workon cv

Attention, maintenant vous devez toujours être sous :

**cv) pi@raspberrypi:~/CNCLASER $**

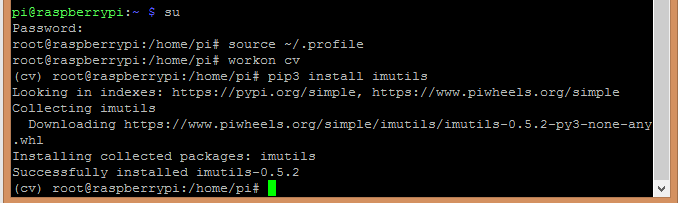
Pour lancer un programme python3, taper le chemin de l’environnement virtuel de python3 :

/root/.virtualenvs/cv/bin/python3.5 /home/pi/CNCLASER/Main\_draft.py

Mais pour cela, via WinSCP, sur l’écran de droite, cliquer droit sur le répertoire « root », puis « propriétés », cocher toutes les cases X, R et W et sur « appliquer récursivement…. », puis « ok »

# **imutils**

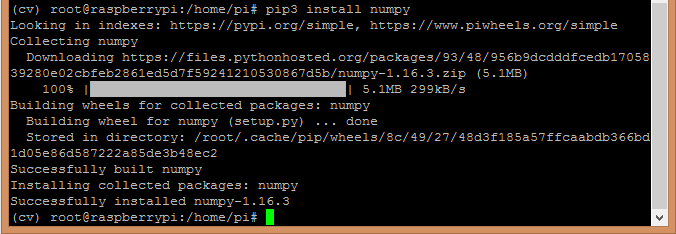
$ pip3 install imutils



# **numpy**

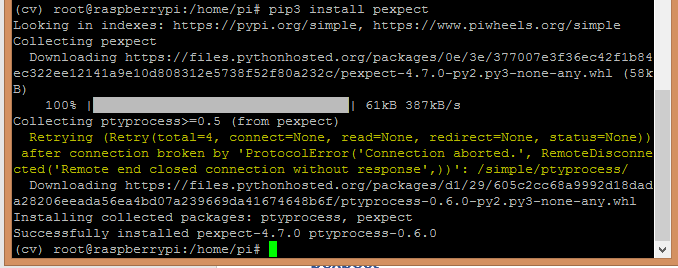
$ pip3 install numpy

Cette installation prend un peu de temps



# **pexpect**

$ pip3 install pexpect



# **PyUSB et cie**

<https://github.com/pyusb/pyusb>

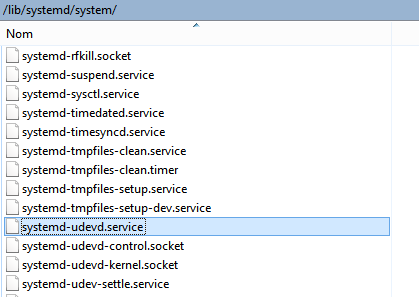
$ pip3 install pyusb

**puis :**

$ sudo apt-get install usbmount

$ pip3 install pyudev

Sous WinSCP, aller dans /lib/systemd/system/systemd-udevd.service pour modifier ce fichier en changeant : MountFlags=slave en MountFlags=shared



Liens utiles

<https://vivekanandxyz.wordpress.com/2017/12/29/detecting-and-automatically-mounting-pendrive-on-raspbian-stretch-lite/>

<https://www.axllent.org/docs/view/auto-mounting-usb-storage/>

Le montage automatique de périphériques USB externes peut s'avérer très pratique, en particulier lorsque vous utilisez des serveurs sans interface graphique (ou sans interface graphique), ou dans mon cas, un Raspberry Pi. Un simple script udev suffit, et suppose que tout périphérique de stockage externe connecté via USB doit être monté automatiquement dans un sous-répertoire de / media.

Créez simplement un fichier /etc/udev/rules.d/11-media-by-label-auto-mount.rules avec ce qui suit:

KERNEL!="sd[a-z][0-9]", GOTO="media\_by\_label\_auto\_mount\_end"

# Import FS infos

IMPORT{program}="/sbin/blkid -o udev -p %N"

# Get a label if present, otherwise specify one

ENV{ID\_FS\_LABEL}!="", ENV{dir\_name}="%E{ID\_FS\_LABEL}"

ENV{ID\_FS\_LABEL}=="", ENV{dir\_name}="usbhd-%k"

# Global mount options

ACTION=="add", ENV{mount\_options}="relatime"

# Filesystem-specific mount options

ACTION=="add", ENV{ID\_FS\_TYPE}=="vfat|ntfs", ENV{mount\_options}="$env{mount\_options},utf8,gid=100,umask=002"

# Mount the device

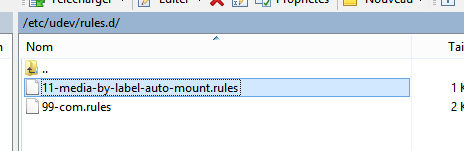
ACTION=="add", RUN+="/bin/mkdir -p /media/%E{dir\_name}", RUN+="/bin/mount -o $env{mount\_options} /dev/%k /media/%E{dir\_name}"

# Clean up after removal

ACTION=="remove", ENV{dir\_name}!="", RUN+="/bin/umount -l /media/%E{dir\_name}", RUN+="/bin/rmdir /media/%E{dir\_name}"

# Exit

LABEL="media\_by\_label\_auto\_mount\_end"



Une fois que vous avez créé et sauvegardé votre script, rechargez udev avec:

$ udevadm control --reload-rules

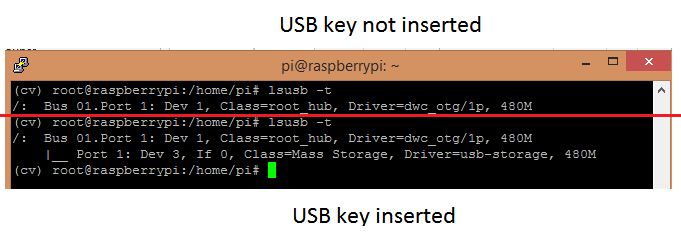
À présent, lorsque vous connectez le périphérique de stockage USB, udev doit créer automatiquement un répertoire sous / media avec l’ID ou l’étiquette du périphérique (si vous en avez attribué un) et monter le périphérique. Si vous déconnectez le périphérique, le répertoire du lecteur (dans / media) sera automatiquement supprimé.

Test sans clé insérée sur le port micro USB du Raspberry PI

$ lsusb -t

Insérer la clé USB, puis taper :

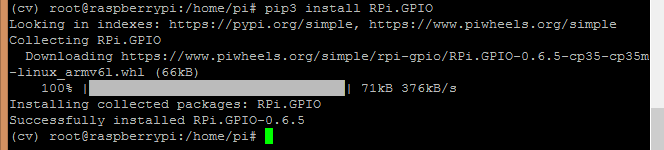
$ lsusb -t



# **IO**

https://sourceforge.net/p/raspberry-gpio-python/wiki/Home/

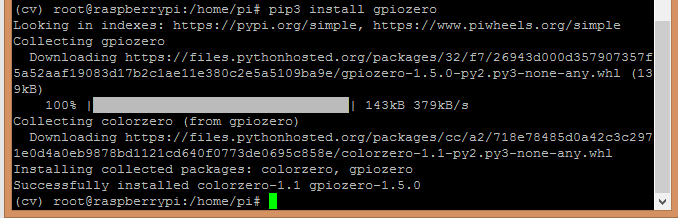
$ pip3 install [RPi.GPIO](https://pypi.python.org/pypi/RPi.GPIO/0.6.3)



et

<https://gpiozero.readthedocs.io/en/stable/installing.html>

$ pip3 install gpiozero



# Afficheur LCD

https://github.com/rm-hull/luma.examples

Activer le port SPI:

$ sudo raspi-config

> Advanced Options > A6 SPI

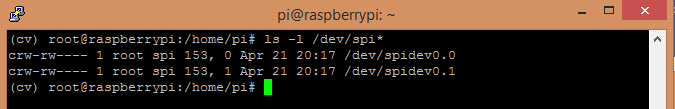
Si raspi-config n'est pas disponible, l'activation du port SPI peut être effectuée manuellement.

Assurez-vous que le pilote de noyau SPI est activé:

$ ls -l /dev/spi\*

crw-rw---- 1 root spi 153, 0 Nov 25 08:32 /dev/spidev0.0

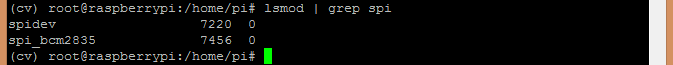
crw-rw---- 1 root spi 153, 1 Nov 25 08:32 /dev/spidev0.1



ou:

$ lsmod | grep spi

spi\_bcm2835 6678 0



Ajoutez ensuite votre utilisateur aux groupes spi et gpio:

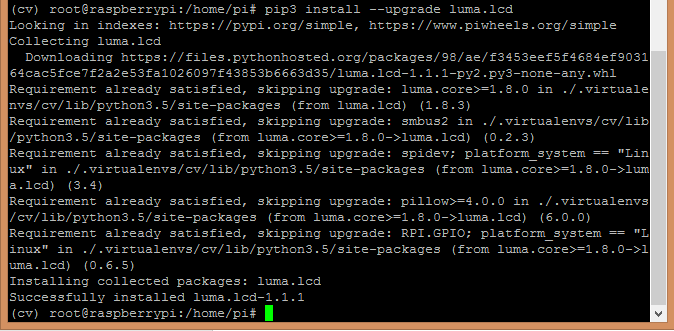
$ usermod -a -G spi pi

$ usermod -a -G gpio pi

Enfin, installez les bibliothèques Luma en utilisant:

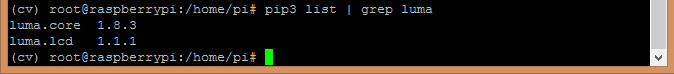
$ sudo apt-get install python-dev python-pip libfreetype6-dev libjpeg-dev build-essential

$ pip3 install --upgrade luma.lcd



Vérifier l’installation :

$ pip list | grep luma



Et un peu de nettoyage :

$ sudo du -h /var/cache/apt/archives

$ sudo apt-get autoclean

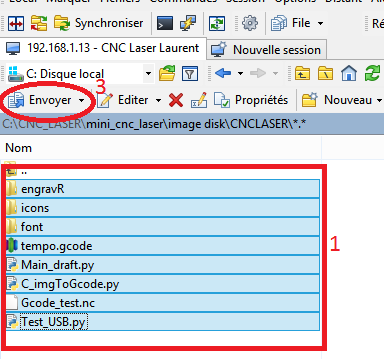
$ sudo apt-get --purge autoremove

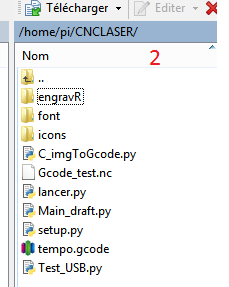
# Copier les fichiers de la CNC laser

Copier les fichiers de l’archive CNCLASER firmware.zip dans «/home/pi »

Pour cela, sous WinCP :

1. sélectionner les fichiers du dossier CNCLASER (1) qui est sur votre PC,
2. Sélectionner le répertoire de déstination (écran de droite sous WinCP) « /home/pi/CNCLASER » (2)
3. Cliquer sur ENVOYER sur l’écran de gauche de WinCP (3)

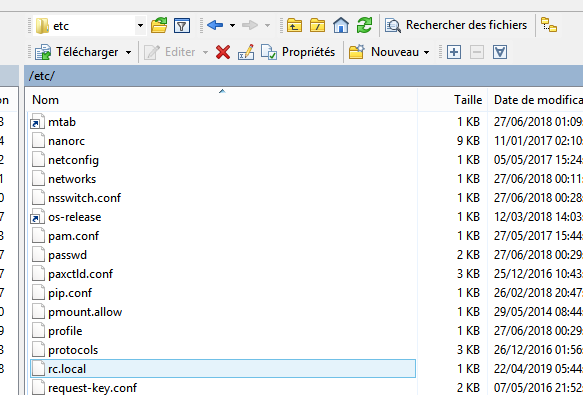




# Configurer le lancement automatique du scripte Python

Sous le répertoire /etc/ et dans le fichier rc.local, copier la ligne :

/root/.virtualenvs/cv/bin/python3.5 /home/pi/CNCLASER/Main\_draft.py &



Contenu du fichier rc.local

#!/bin/sh -e

#

# rc.local

#

# This script is executed at the end of each multiuser runlevel.

# Make sure that the script will "exit 0" on success or any other

# value on error.

#

# In order to enable or disable this script just change the execution

# bits.

#

# By default this script does nothing.

/root/.virtualenvs/cv/bin/python3.5 /home/pi/CNCLASER/Main\_draft.py &

# Print the IP address

\_IP=$(hostname -I) || true

if [ "$\_IP" ]; then

printf "My IP address is %s\n" "$\_IP"

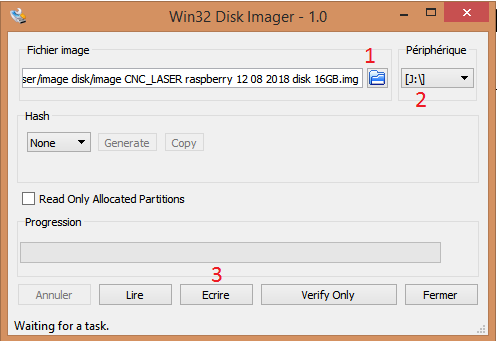
fi

exit 0

Redémarrer le Raspberry Pi et vérifier sur l’écran TV que le scripte est bien démarré. « stepper configured » doit apparaitre, + d’autres infos.

L’écran LCD doit afficher l’IHM.

# Install the Rasbian stretch disk image, by copying the disk image to the SD card.

* Open Windiskimager
* Select the image to copy on the SD card (1)
* 
* Select the device to burn (2)
* Click on Write (3)

Copy the SSH setting files to the SD card:

1. Copy / create an empty file named: "ssh" without extension
2. Copy / create a file named "wpa\_supplicant.conf" containing:

ctrl\_interface=DIR=/var/run/wpa\_supplicant GROUP=netdev

network={

ssid="*name of your ADSL box*"

psk="*code like xx56…….R458…"*

key\_mgmt=WPA-PSK

}

Put the SD card in the raspberry pi (RPI)

Connect a monitor to the HDMI port.

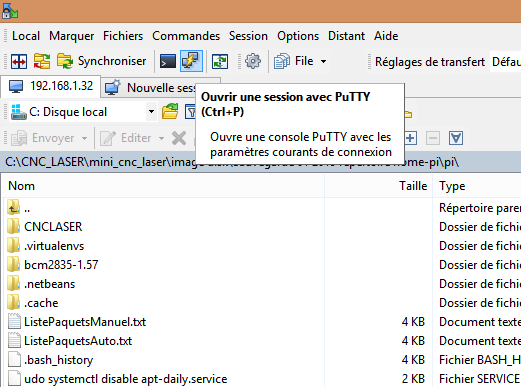
Light the raspberry pi.

Make sure you are close to wifi internet access.

Watch what scrolls on the screen and find: "My IP address is 192.168.1.xx"

Note this address to connect to it in SSH.

Connect to the RPI in ssh via WinSCP (see the associated tutorial)



The login is "pi"

The password is "raspberry"

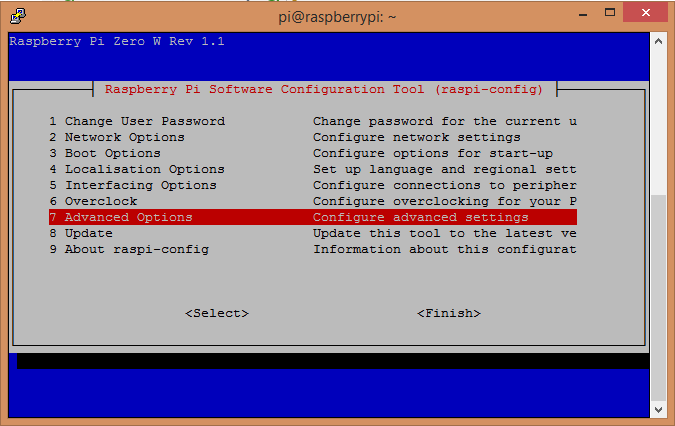
If your Raspbian installation is brand new, you will need to expand the partition to use all free space. For that :

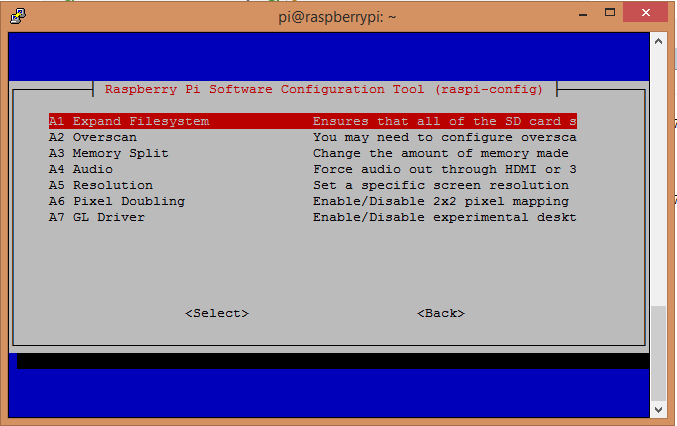
$ sudo raspi-config



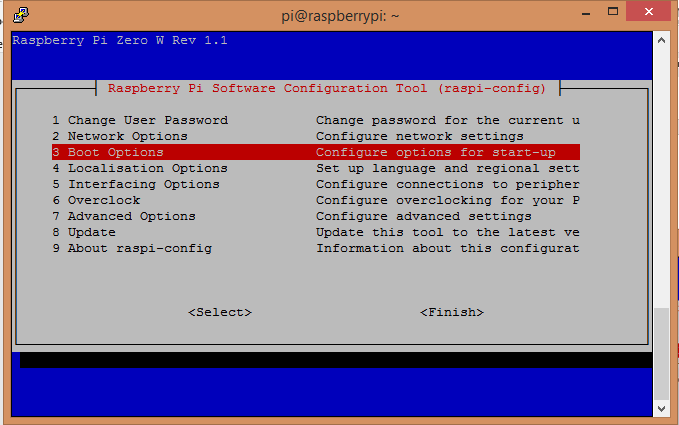
Raspi-config: Expand Filesystem

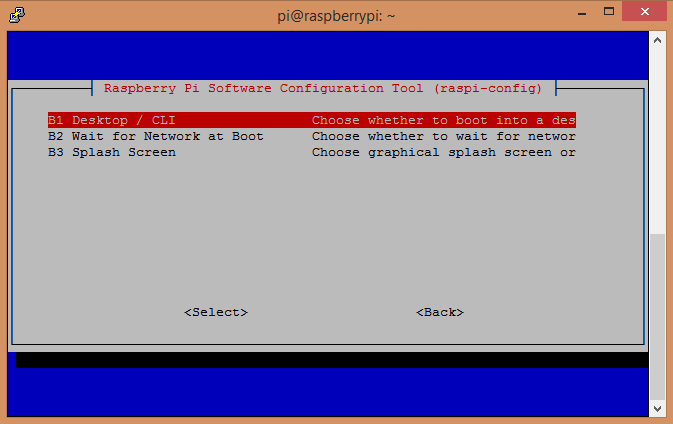
In the menu, go to the option "Expand Filesystem" and press enter and confirm.

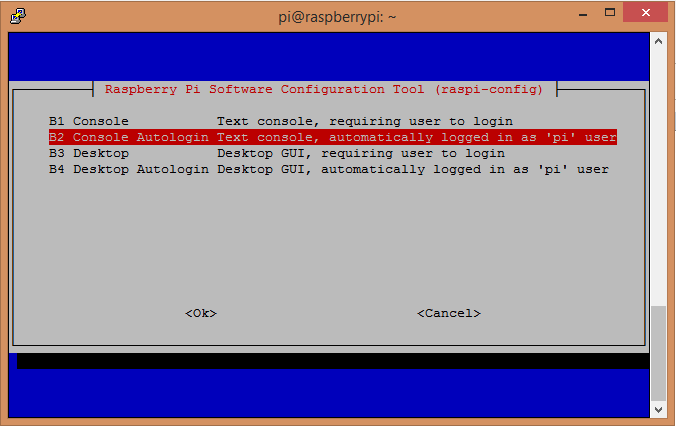


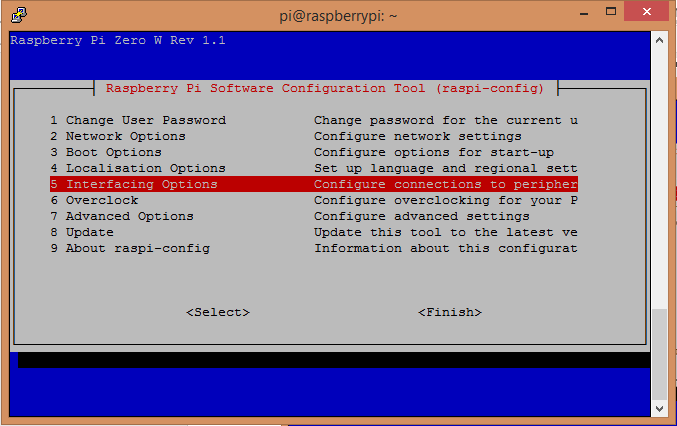


In Boot options, configure the raspberry pi to start without login.







Interface configuration:

Then click SPI interface, and choose ok.

Finally restart your Raspberry:

$ sudo reboot

Optionally create super user password:

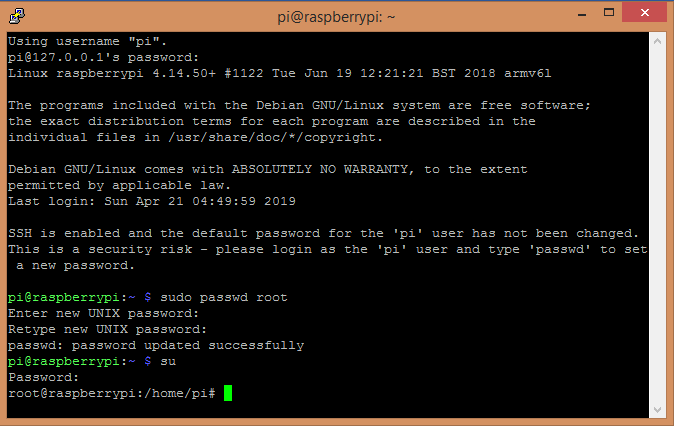
$ sudo passwd root

Enter the password: "rpi"

To access root, type

$ su

Then the password

The command line should be like this now:

Open a Putty console to install the following packages:

Updating packages:

$ sudo apt-get update

$ sudo apt upgrade

# **Install Virtual Environment and Python 3.5:**

Python3.5 must be installed:

$ sudo apt-get install python3.5

Installation of pip and pip3 :

$ curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py

$ python get-pip.py

$ apt-get -y install python3-pip

$ apt-get update

Only 3 applications must be global: python3.5, pip and virtualenv. Is :

$ sudo pip install virtualenv

Or

$ sudo apt-get install virtualenv

And an update:

$ sudo pip install --upgrade virtualenv

**creating a folder for your projects/environments**

$ mkdir CNCLASER

$ cd CNCLASER

**creating environment**

<https://virtualenvwrapper.readthedocs.io/en/latest/command_ref.html>

$ sudo pip**3** install virtualenv virtualenvwrapper

$ sudo rm -rf ~/.cache/pip

Type the following commands:

$ echo -e "\n# virtualenv and virtualenvwrapper" >> ~/.profile

$ echo "export WORKON\_HOME=$HOME/.virtualenvs" >> ~/.profile

$ echo "export VIRTUALENVWRAPPER\_PYTHON=/usr/bin/python3.5" >> ~/.profile

$ echo "source /usr/local/bin/virtualenvwrapper.sh" >> ~/.profile

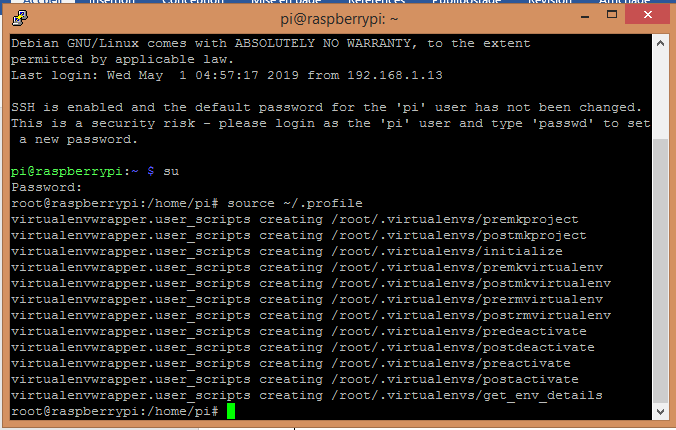
Close the terminal and reopen it

$ su

And enter the password « rpi »

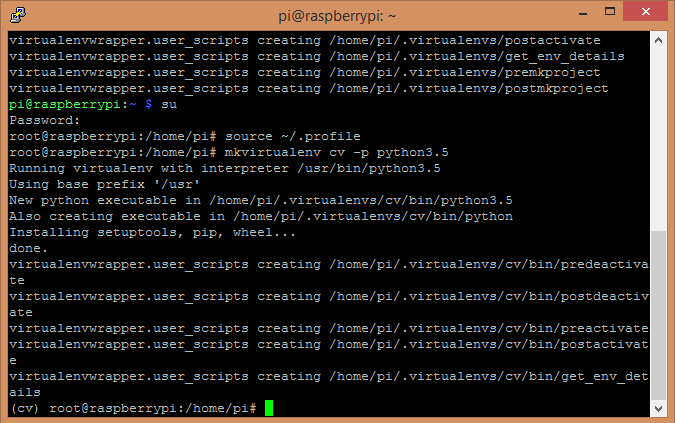
And  :

$ source ~/.profile



$ mkvirtualenv cv -p python3.5

Some time is needed to install the tools of the environment.



Then every time changes need to be made to the installation raspberry pi

$ source ~/.profile

$ workon cv

Attention, now you must always be under:

**cv) pi@raspberrypi:~/CNCLASER $**

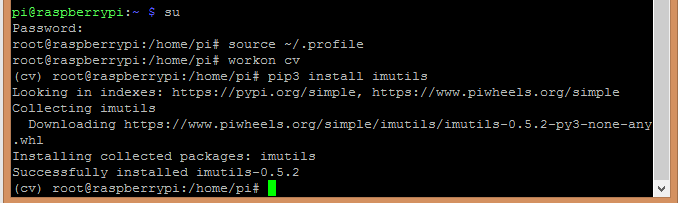
To launch a python3 program, type the path of the python3 virtual environment:

/root/.virtualenvs/cv/bin/python3.5 /home/pi/CNCLASER/Main\_draft.py

But for this, via WinSCP, on the right screen, right click on the "root" directory, then "properties", tick all the boxes X, R and W and "apply recursively .... », Then« ok »

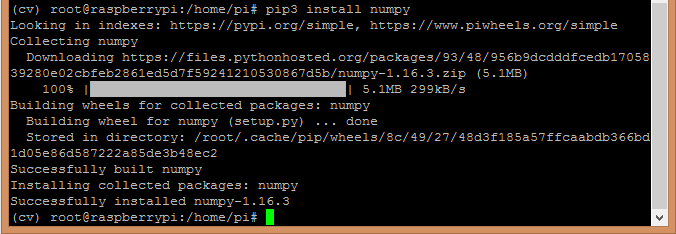
# **imutils**

$ pip3 install imutils



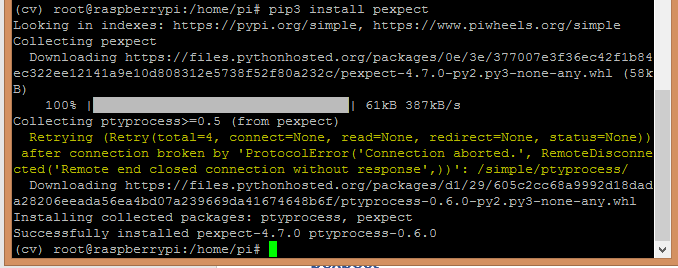
# **numpy**

$ pip3 install numpy



# **pexpect**

$ pip3 install pexpect



# **PyUSB et cie**

<https://github.com/pyusb/pyusb>

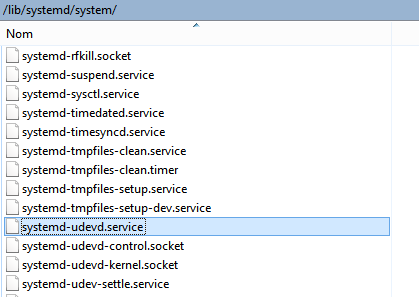
$ pip3 install pyusb

**And :**

$ sudo apt-get install usbmount

$ pip3 install pyudev

Under WinSCP, go to /lib/systemd/system/systemd-udevd.service to modify this file by changing: MountFlags = slave to MountFlags = shared



Liens utiles

<https://vivekanandxyz.wordpress.com/2017/12/29/detecting-and-automatically-mounting-pendrive-on-raspbian-stretch-lite/>

<https://www.axllent.org/docs/view/auto-mounting-usb-storage/>

Auto-mounting external USB devices can be very handy, especially when using headless (no GUI) servers, or in my case a Raspberry Pi. A simple udev script is all that is needed, and assumed that any external storage device connected via USB will need to be mounted automatically in a subdirectory of /media

Simply create a file /etc/udev/rules.d/11-media-by-label-auto-mount.rules with the following:

KERNEL!="sd[a-z][0-9]", GOTO="media\_by\_label\_auto\_mount\_end"

# Import FS infos

IMPORT{program}="/sbin/blkid -o udev -p %N"

# Get a label if present, otherwise specify one

ENV{ID\_FS\_LABEL}!="", ENV{dir\_name}="%E{ID\_FS\_LABEL}"

ENV{ID\_FS\_LABEL}=="", ENV{dir\_name}="usbhd-%k"

# Global mount options

ACTION=="add", ENV{mount\_options}="relatime"

# Filesystem-specific mount options

ACTION=="add", ENV{ID\_FS\_TYPE}=="vfat|ntfs", ENV{mount\_options}="$env{mount\_options},utf8,gid=100,umask=002"

# Mount the device

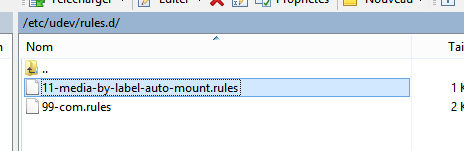
ACTION=="add", RUN+="/bin/mkdir -p /media/%E{dir\_name}", RUN+="/bin/mount -o $env{mount\_options} /dev/%k /media/%E{dir\_name}"

# Clean up after removal

ACTION=="remove", ENV{dir\_name}!="", RUN+="/bin/umount -l /media/%E{dir\_name}", RUN+="/bin/rmdir /media/%E{dir\_name}"

# Exit

LABEL="media\_by\_label\_auto\_mount\_end"



Once you have created your script and saved it, reload udev with:

$ udevadm control --reload-rules

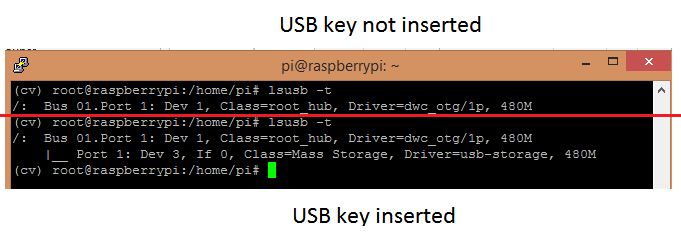
Now, when you connect the USB storage device, udev should automatically create a directory under /media with either the device ID, or device label (if you have assigned one), and mount the device. If you disconnect the device, the drive directory (in /media) will be automatically removed.

Keyless test inserted on the Raspberry PI micro USB port

$ lsusb -t

Insert the USB key, then type:

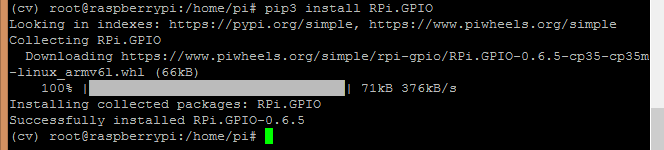
$ lsusb -t



# **IO**

https://sourceforge.net/p/raspberry-gpio-python/wiki/Home/

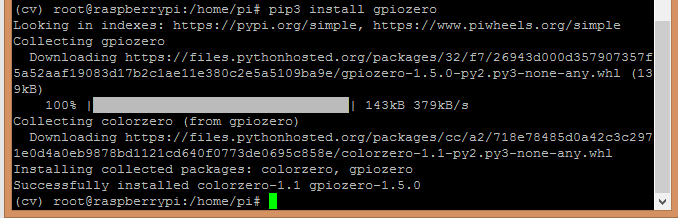
$ pip3 install [RPi.GPIO](https://pypi.python.org/pypi/RPi.GPIO/0.6.3)



et

<https://gpiozero.readthedocs.io/en/stable/installing.html>

$ pip3 install gpiozero



# Afficheur LCD

<https://github.com/rm-hull/luma.examples>

Enable the SPI port:

$ sudo raspi-config

> Advanced Options > A6 SPI

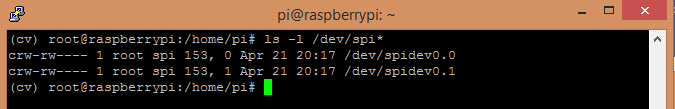
If raspi-config is not available, enabling the SPI port can be done [manually](http://elinux.org/RPiconfig#Device_Tree).

Ensure that the SPI kernel driver is enabled:

$ ls -l /dev/spi\*

crw-rw---- 1 root spi 153, 0 Nov 25 08:32 /dev/spidev0.0

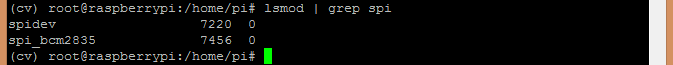
crw-rw---- 1 root spi 153, 1 Nov 25 08:32 /dev/spidev0.1



or:

$ lsmod | grep spi

spi\_bcm2835 6678 0



Then add your user to the spi and gpio groups:

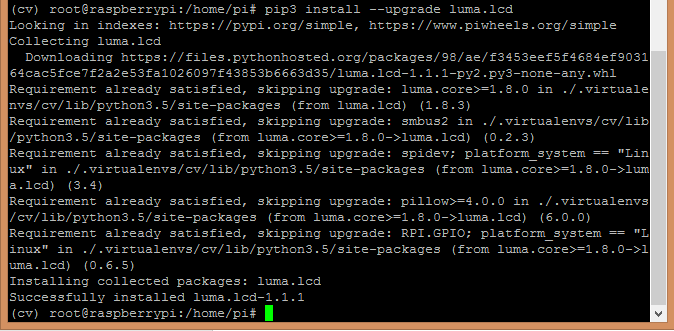
$ usermod -a -G spi pi

$ usermod -a -G gpio pi

Finally, install the luma libraries using:

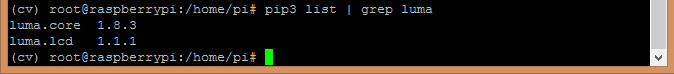
$ sudo apt-get install python-dev python-pip libfreetype6-dev libjpeg-dev build-essential

$ pip3 install --upgrade luma.lcd



Check the installation:

$ pip list | grep luma



And a little cleaning:

$ sudo du -h /var/cache/apt/archives

$ sudo apt-get autoclean

$ sudo apt-get --purge autoremove

Copy laser CNC files

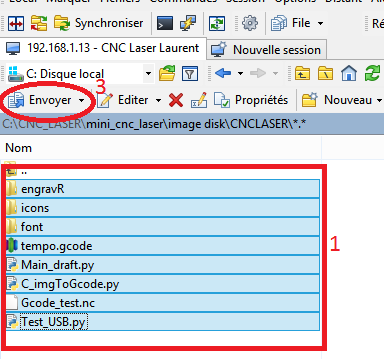
Copy the files of the archive CNCLASER firmware.zip in "/ home / pi"

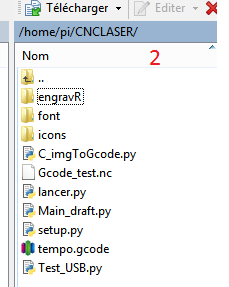
For this, under WinCP:

a) select the files from the folder CNCLASER (1) which is on your PC,

b) Select destination directory (right screen under WinCP) "/ home / pi / CNCLASER" (2)

c) Click SEND on the left screen of WinCP (3)

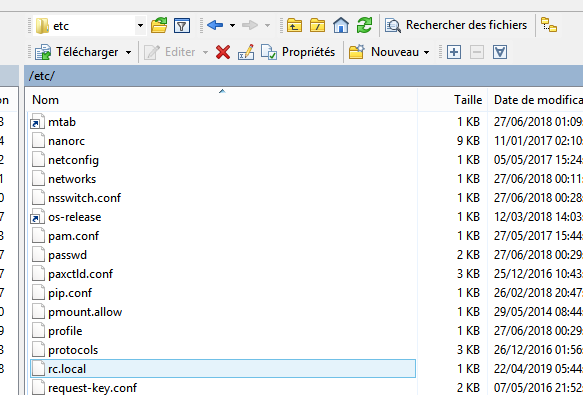




# Configure the automatic launch of the Python script

Under the / etc / directory and in the rc.local file, copy the line:

/root/virtualenvs/cv/bin/python3.5 /home/pi/CNCLASER/Main\_draft.py &



Contents of the rc.local file

#!/bin/sh -e

#

# rc.local

#

# This script is executed at the end of each multiuser runlevel.

# Make sure that the script will "exit 0" on success or any other

# value on error.

#

# In order to enable or disable this script just change the execution

# bits.

#

# By default this script does nothing.

/root/.virtualenvs/cv/bin/python3.5 /home/pi/CNCLASER/Main\_draft.py &

# Print the IP address

\_IP=$(hostname -I) || true

if [ "$\_IP" ]; then

printf "My IP address is %s\n" "$\_IP"

fi

exit 0

Restart the Raspberry Pi and check on the TV screen that the script is started. "Stepper configured" should appear, + more info.

The LCD should display the HMI.