

Steps for Controlling Autonomous Tracking System

modified from Talha Kose (May 17, 2022)

Screenshots/examples are run on Windows

1. Download and install UtorVPN, Cisco.

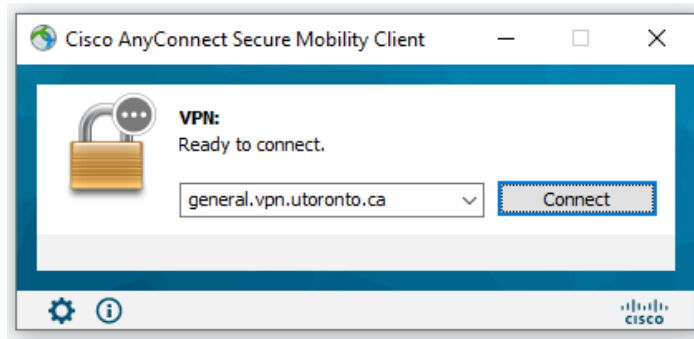
Here is the link for UtorVPN. Depending on your device, select the proper operating system, install The Cisco AnyConnect Client with the proper settings.

<https://onereach.library.utoronto.ca/ic-faq-categories/utorvpn>

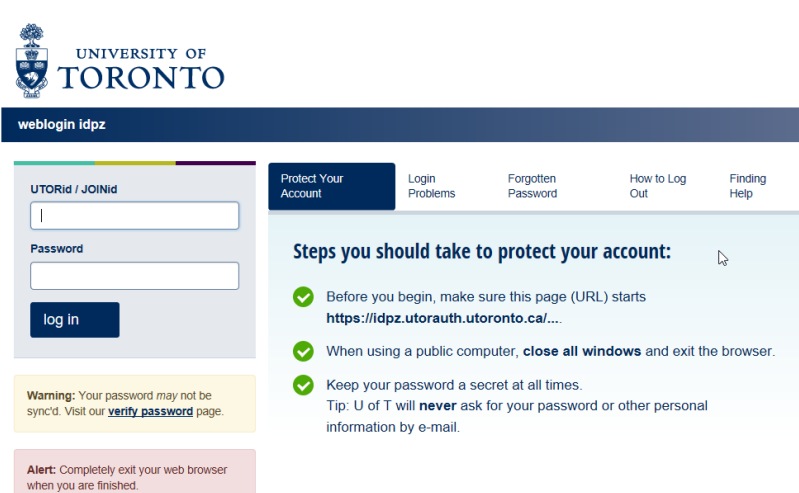
2. Open VPN and connect via your UTORid and password.

Open the AnyConnect Client.

Enter or select **general.vpn.utoronto.ca** and click **Connect**

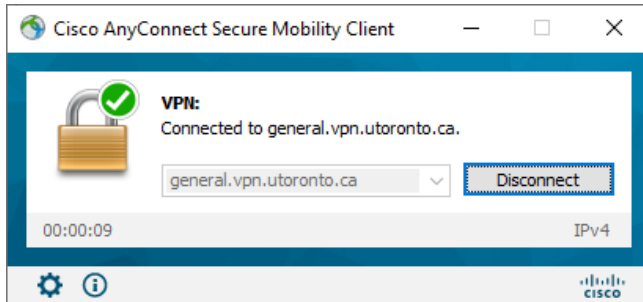


An authentication window will open for the group “utorvpn”



Enter your UTORid and password. Click **OK**.

Your computer will now notify you that it is connected to the VPN.



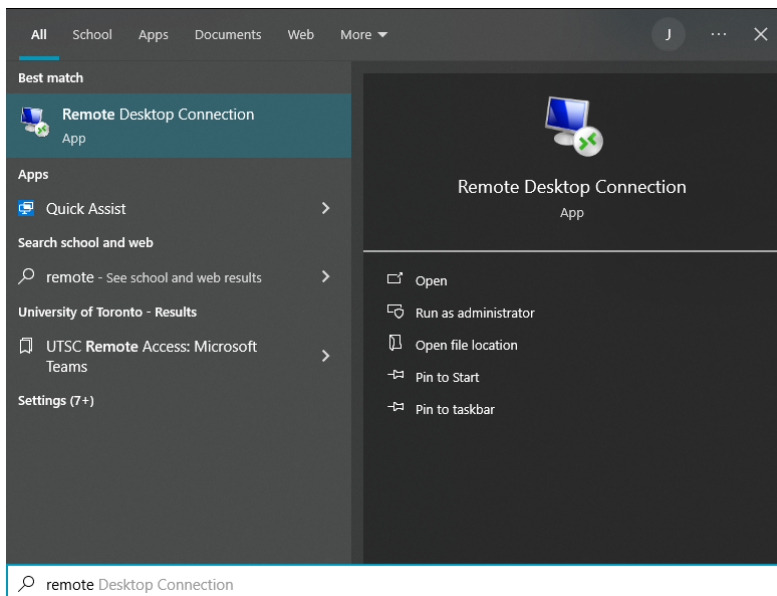
NOTE: If you are on campus and connected to UofT Wifi network or if you use a computer in the lab registered to local network via ethernet cable, you don't need to run the VPN. You can skip the first two steps and start from the Step 3 below.

- However, I have found that I still need to connect to the VPN when using the wireless network in ESC

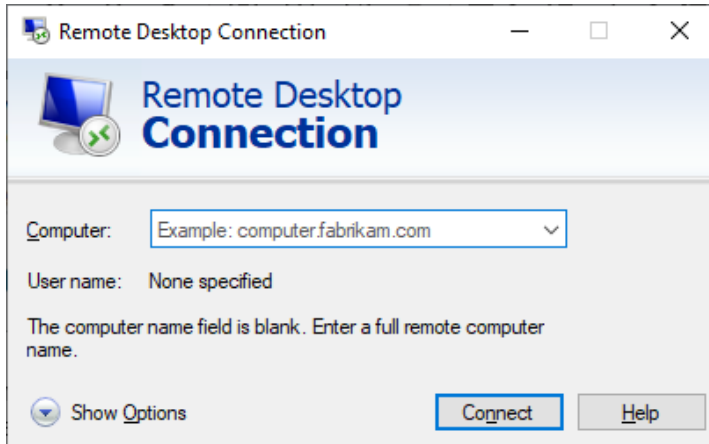
3. Windows -> Remote Desktop Connection

In order to connect to the Raspberry Pi of the imaging system, you need to run 'Remote Desktop Connection'

Press the Windows button and search "remote" in the search bar.



Click on "Remote Desktop Connection"



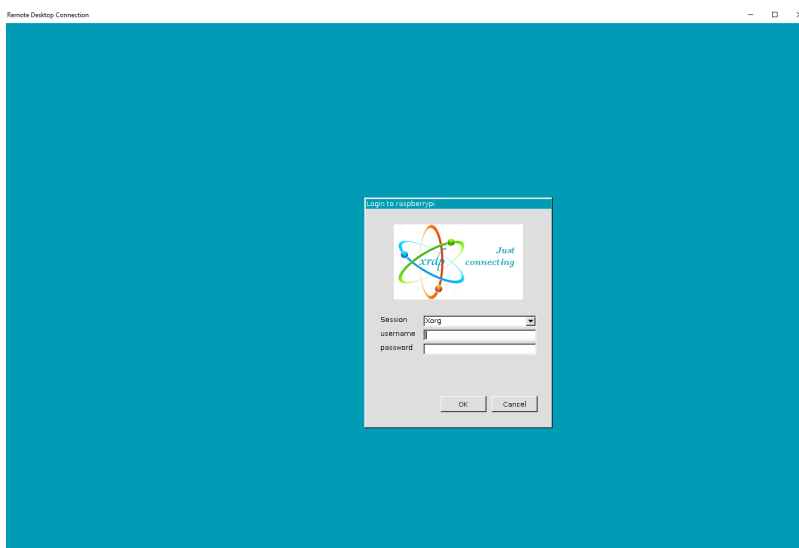
In the 'Computer' field, enter the IP address of the RPi and click connect.

IP: see Slack!

- Once you have connected to the RPi, the IP address will be saved for future connections.

4. Login to the Raspberry Pi.

After you connect, a new window will pop up and prompt you to log in to the RPi. This window is where you will access the imaging system.

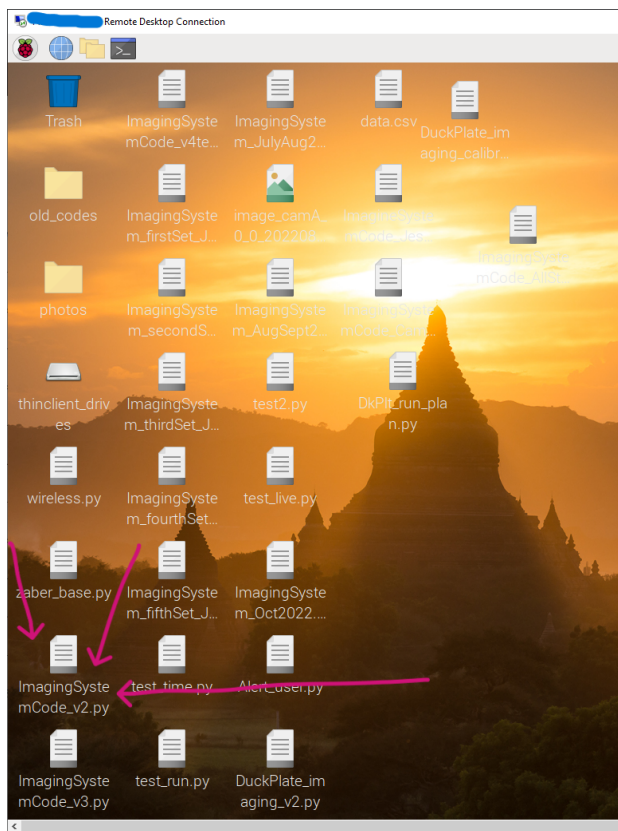


User and Password: see Slack!

- To quickly change between the remote connection and your computer, you can switch windows by pressing the 'ALT' and 'Tab' buttons.

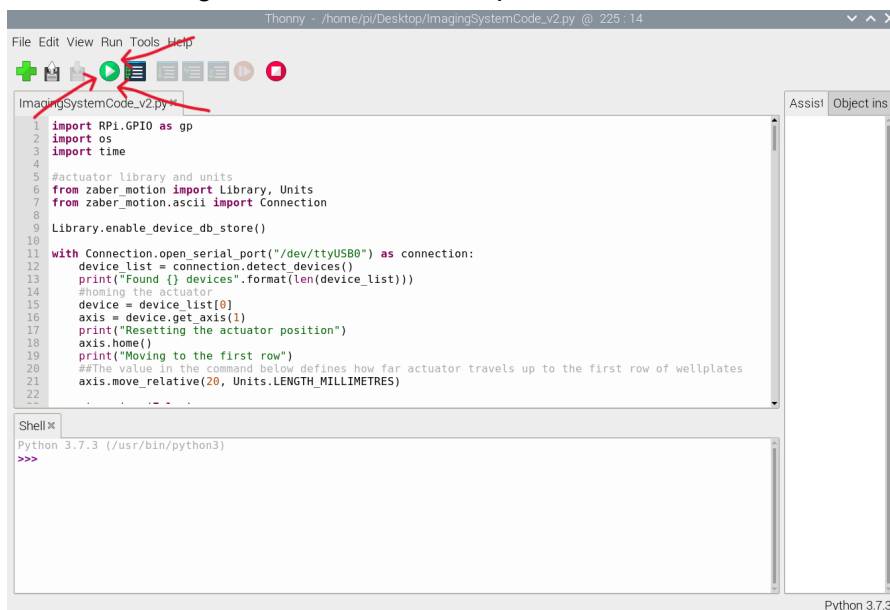
5. Open and run the python code controlling the imaging system.

A copy of the code is on the Desktop, titled 'ImagingSystemCode_v2.py'
Another copy of this code is located on GitHub.



(yes, the desktop is currently a mess)

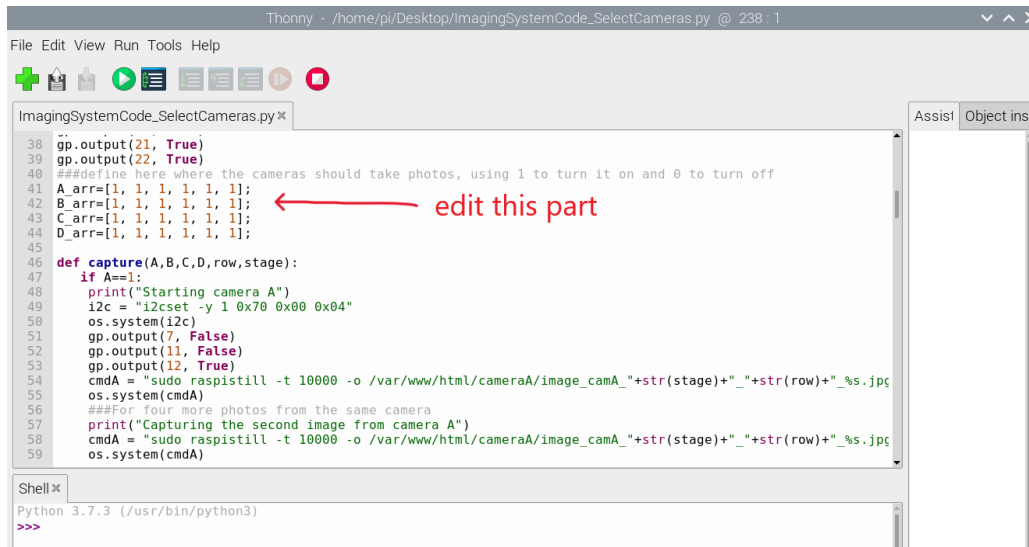
Double click this file, or whatever your code file is and open it in Thonny. To run the code, click the green “Run current script” button.



After you click run, you can monitor the progress of the code in Shell panel below in the Thonny window.

5a. Editing the python code (some notes)

Use “ImagingSystemCode_SelectCameras.py” to select specific cameras and/or rows to be imaged



```
Thonny - /home/pi/Desktop/ImagingSystemCode_SelectCameras.py @ 238 : 1
File Edit View Run Tools Help
+ [Icons]
ImagingSystemCode_SelectCameras.py x
38 gp.output(21, True)
39 gp.output(22, True)
40 ##define here where the cameras should take photos, using 1 to turn it on and 0 to turn off
41 A_arr=[1, 1, 1, 1, 1, 1];
42 B_arr=[1, 1, 1, 1, 1, 1];
43 C_arr=[1, 1, 1, 1, 1, 1];
44 D_arr=[1, 1, 1, 1, 1, 1];
45
46 def capture(A, B, C, D, row, stage):
47     if A==1:
48         print("Starting camera A")
49         i2c = "i2cset -y 1 0x70 0x00 0x04"
50         os.system(i2c)
51         gp.output(7, False)
52         gp.output(11, False)
53         gp.output(12, True)
54         cmdA = "sudo raspistill -t 10000 -o /var/www/html/cameraA/image_camA_"+str(stage)+"_"+str(row)+"_%.jpg"
55         os.system(cmdA)
56         ##for four more photos from the same camera
57         print("Capturing the second image from camera A")
58         cmdA = "sudo raspistill -t 10000 -o /var/www/html/cameraA/image_camA_"+str(stage)+"_"+str(row)+"_%.jpg"
59         os.system(cmdA)
Shell x
Python 3.7.3 (/usr/bin/python3)
>>>
```

- Some comments in ImagingSystemCode_v2.py to understand
- axis.home() - resets axis position
- Then moves to first row of where experiment starts
- Axis.move_relative(x, Units.LENGTH_MILLIMETRES) -> x = # mm moving
- If more plates, need to change x
 - X=20 is for first row of first transparent stage
- Do not touch gp.output()
- First def main() - can adjust camera parameters
- Select parts to take more/less photos - press ALT+3 -> comment in entire section
 - ALT+4 -> comment out
 - Other commands tell where to save, etc.
- Range = how many rows on stage
 - For i in range (0, x) -> x is number of rows (max. 5)
- If i == (x-1): break -- so if 5 rows, this needs to be 4
- Second stage: axis position is 210 mm, max. 5 rows
- Third stage: axis position is 205 mm, max. 4 rows
- Total 14 rows, 8 plates in each row = total 112 well plates
 - 10752 wells if 112x 96-well plates
- 'Run current script' to run the code
- **Do not change going home part!**

6. Access the captured photos.

All the files (images or videos) are stored in the following location.

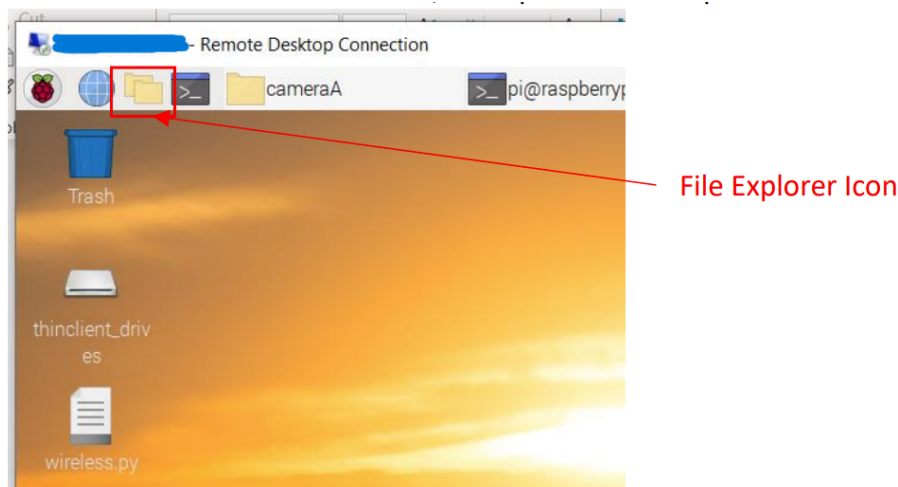
“/var/www/html/cameraA”

“/var/www/html/cameraB”

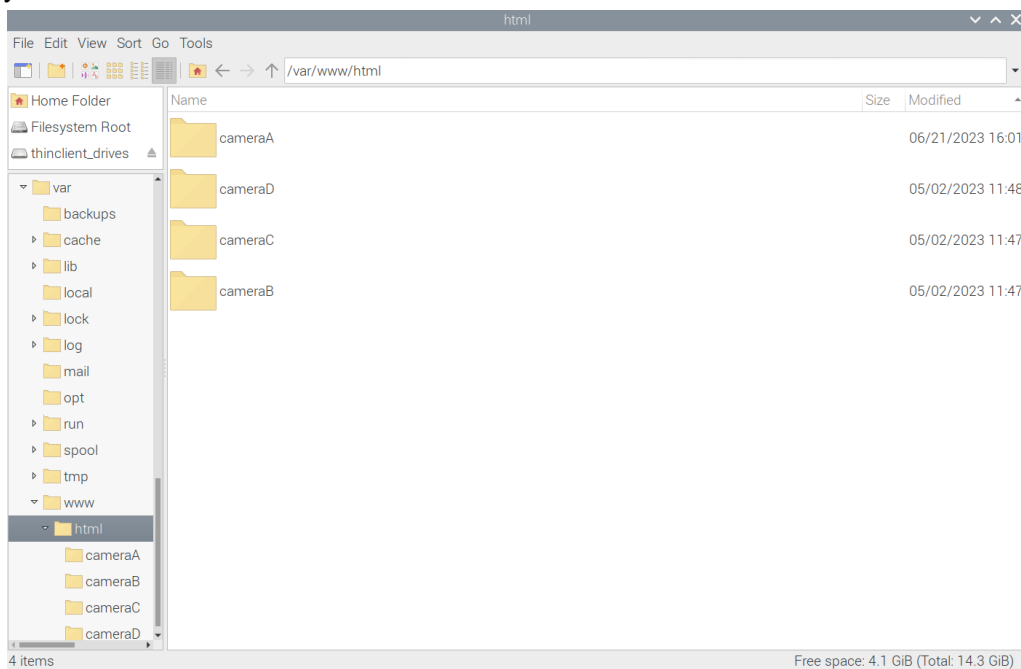
“/var/www/html/cameraC”

“/var/www/html/cameraD”

To navigate this location, you can use File Explorer which can be reached from the Icon on the Task Bar. Click on the folders icon, and open the File Explorer.



Using the left panel of the File Explorer Window, navigate through var>www>html where you will find these four folders shown below:



7. Copying the files from RPi to your local computer

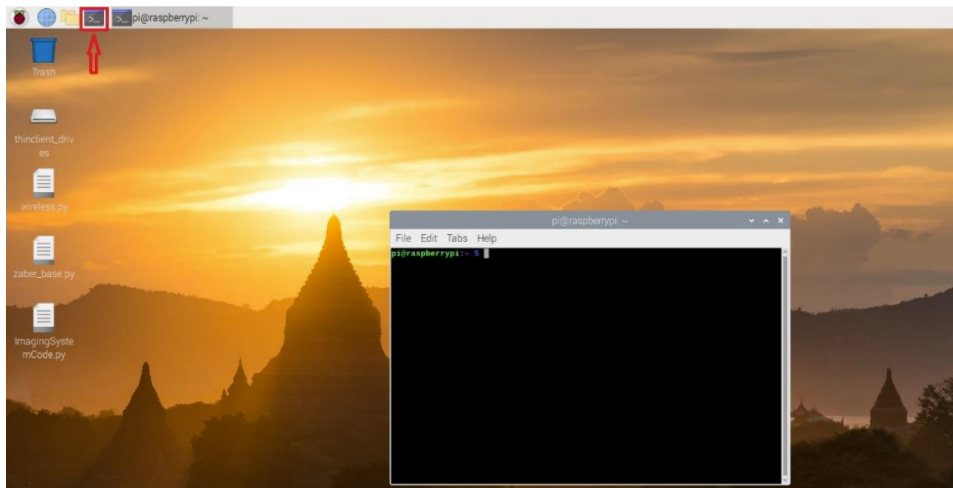
Transferring the stored files in cameraA, cameraB, cameraC and cameraD folders is simply a copy-paste action. Select the files you want to transfer to your local device, right click and copy the files. Then, minimize the remote desktop window, and navigate to the destination folder on your local device. Then, right click and paste. File transfer will begin right after. Wait for the transfer to complete.

- a. Select the photos in RPi folder.
 - b. Right click
 - c. Copy
 - d. Minimize your remote desktop connection window
 - If you are in full screen mode, use the blue bar on the top to minimize the remote desktop connection window to reach your folders on your local device.
 - e. Right click and paste the files to the location you wanted in your local device.
- *To quickly change between the remote connection and your computer, you can switch windows by pressing the 'ALT' and 'Tab' buttons.*

(Optional) How to delete photos in RPi after copying it to your local computer

RPi has limited space for the photos captured via the autonomous tracking system, which is around 10 GB. Assuming you are conducting an experiment in full capacity (with 112 well-plates), it requires 280 photos (56×5 – one camera is responsible for imaging 2 well-plates and 5 images capture via a camera to eliminate vibration errors) to be stored per day. Based on our observations, a raw photo covers 6-7 MB of free space, which makes 1.6-2 GB of data per day. That means you have to unload the photos taken every 4 days if you use the system at full capacity. Hence, after making sure you copied the photos successfully to your local computer or your local server, you have to delete all the images taken for your experiment. Here you can find how to do so.

- a. Open the RPi terminal using the button on the task bar.



- b. Write or copy the following line and press Enter to remove all the files saved in the Camera C folder.

```
sudo rm /var/www/html/cameraC/*.jpg
```

Please note that the *.jpg part in this line means “any file name with jpg extensions”. If you want to delete a certain image (say its name is image20220507-165357.jpg), you should edit the line as the following:

```
sudo rm /var/www/html/cameraC/image20220507-165357.jpg
```

Similarly, you have to run this code for each camera folder (Camera A, CameraB, CameraC and CameraD) by changing the letter C with A, B and D.

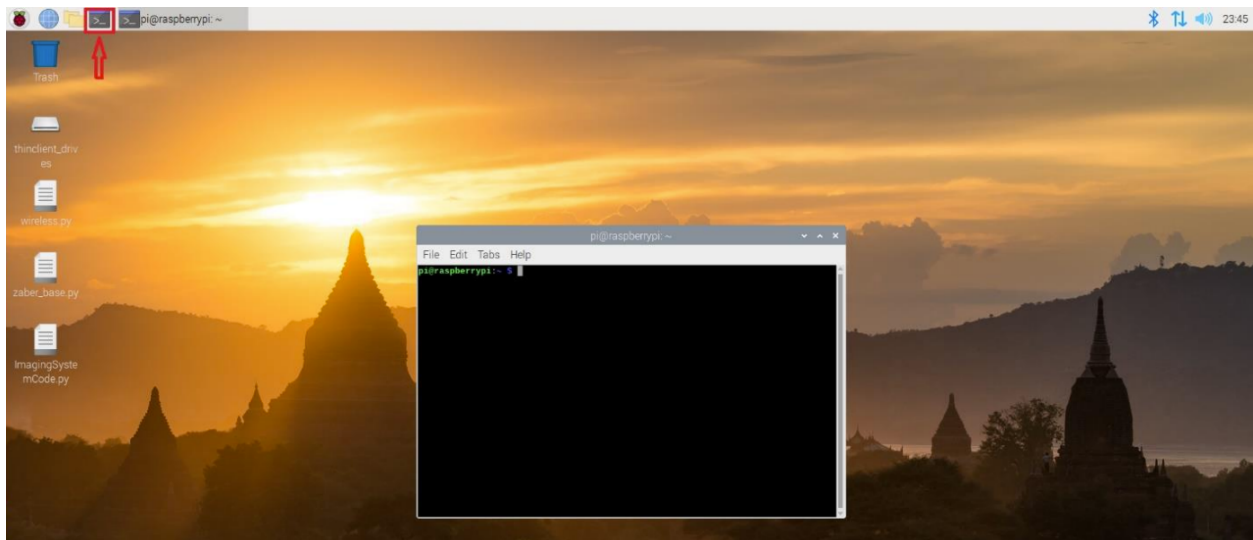
Note: if you change/carry the folders of the cameras, you have to make sure the “/var/www/html/cameraC/” part of the code is updated accordingly.

After pressing enter and running the deletion, there will not be an update on the terminal. However, if you check the folder, you will see all images are gone.

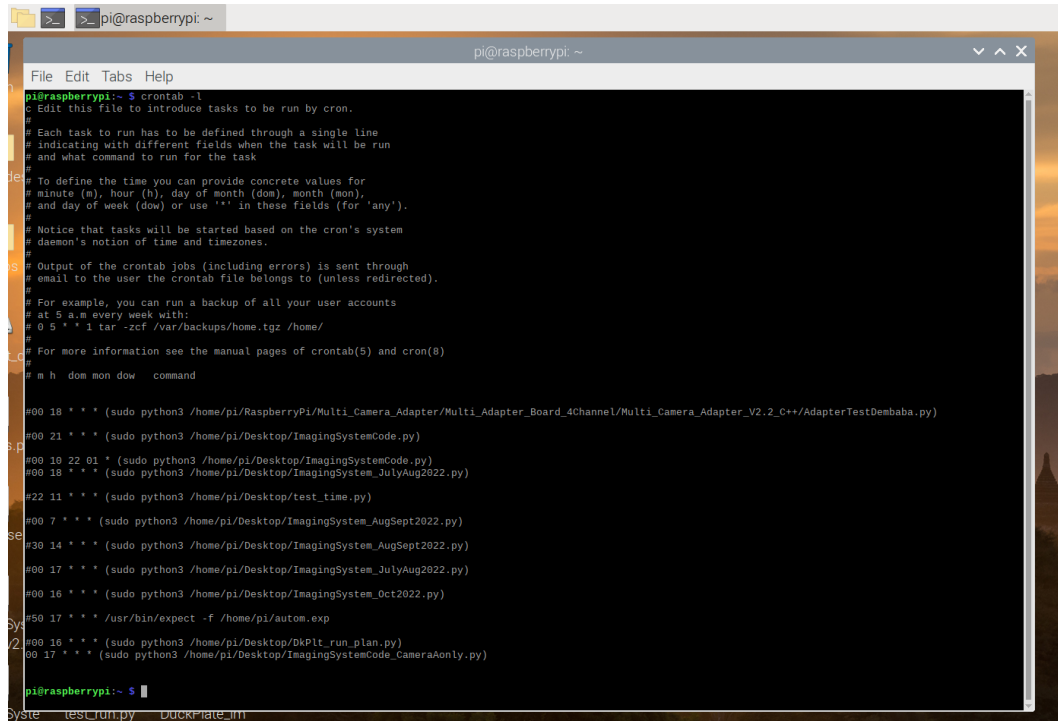
8. Once you are done with the file transfer and imaging system, close the remote desktop window.

Scheduling the code

In order to set a pre-defined date to run the code automatically, we must run crontab code from the Raspberry Pi Terminal. Raspberry Pi Terminal can be opened using the icon on the task bar.



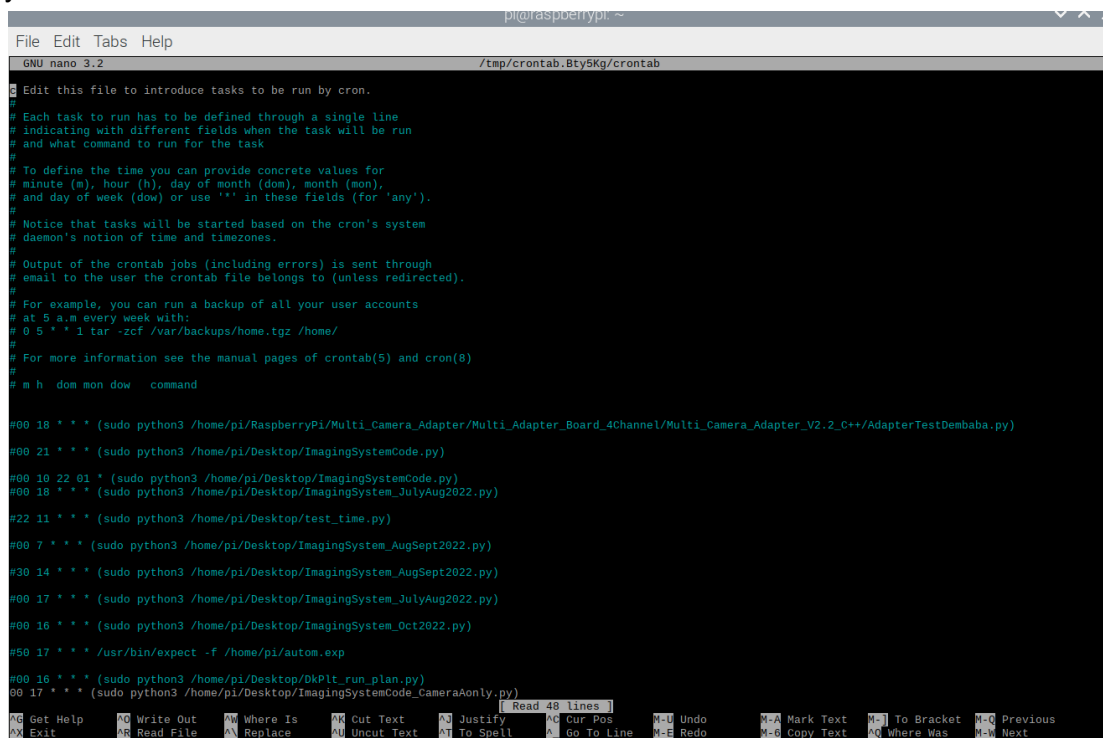
On the terminal, write “crontab -l” and press enter to view any scheduled task for the Raspberry Pi. Note that any line starting with “#” is commented out, which means it is inactive. In the image below, all lines are commented out, hence, there is no active scheduled task for this RPi system.

A screenshot of a terminal window on a Raspberry Pi. The window title is 'pi@raspberrypi: ~'. The terminal shows the output of the 'crontab -l' command, displaying the current crontab configuration. The file contains several commented-out tasks and one active task. The active task is at line 18: '#00 18 * * * (sudo python3 /home/pi/RaspberryPi/Multi_Camera_Adapter/Multi_Adapter_Board_4Channel/Multi_Camera_Adapter_V2.2_C++/AdapterTestDembaba.py)'. The terminal output is as follows:

```
pi@raspberrypi:~$ crontab -l
#
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
#
#00 18 * * * (sudo python3 /home/pi/RaspberryPi/Multi_Camera_Adapter/Multi_Adapter_Board_4Channel/Multi_Camera_Adapter_V2.2_C++/AdapterTestDembaba.py)
#00 21 * * * (sudo python3 /home/pi/Desktop/ImagingSystemCode.py)
#00 10 22 01 * (sudo python3 /home/pi/Desktop/ImagingSystemCode.py)
#00 18 * * * (sudo python3 /home/pi/Desktop/ImagingSystem_JulyAug2022.py)
#22 11 * * * (sudo python3 /home/pi/Desktop/test_time.py)
#00 7 * * * (sudo python3 /home/pi/Desktop/ImagingSystem_AugSept2022.py)
#30 14 * * * (sudo python3 /home/pi/Desktop/ImagingSystem_AugSept2022.py)
#00 17 * * * (sudo python3 /home/pi/Desktop/ImagingSystem_JulyAug2022.py)
#00 16 * * * (sudo python3 /home/pi/Desktop/ImagingSystem_Oct2022.py)
#50 17 * * * /usr/bin/expect -f /home/pi/autom.exp
#00 16 * * * (sudo python3 /home/pi/Desktop/DkPlt_run_plan.py)
#00 17 * * * (sudo python3 /home/pi/Desktop/ImagingSystemCode_CameraAonly.py)

pi@raspberrypi:~$
```

To edit the crontab tasks, write “crontab -e” command on the Pi Terminal. Once you hit enter, you’ll see the crontab becomes editable.

A screenshot of the nano text editor editing the crontab file. The window title is 'pi@raspberrypi: ~'. The editor shows the same crontab content as the previous screenshot, but the final line is now active (white text on a dark background). The active task is at line 18: '00 18 * * * (sudo python3 /home/pi/RaspberryPi/Multi_Camera_Adapter/Multi_Adapter_Board_4Channel/Multi_Camera_Adapter_V2.2_C++/AdapterTestDembaba.py)'. The editor interface includes a menu bar at the top and a status bar at the bottom. The status bar shows 'Read 48 lines' and various keyboard shortcuts. The terminal output is as follows:

```
pi@raspberrypi:~$ crontab -e
GNU nano 3.2 /tmp/crontab.Bty5Kg/crontab
#
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
#
00 18 * * * (sudo python3 /home/pi/RaspberryPi/Multi_Camera_Adapter/Multi_Adapter_Board_4Channel/Multi_Camera_Adapter_V2.2_C++/AdapterTestDembaba.py)
#00 21 * * * (sudo python3 /home/pi/Desktop/ImagingSystemCode.py)
#00 10 22 01 * (sudo python3 /home/pi/Desktop/ImagingSystemCode.py)
#00 18 * * * (sudo python3 /home/pi/Desktop/ImagingSystem_JulyAug2022.py)
#22 11 * * * (sudo python3 /home/pi/Desktop/test_time.py)
#00 7 * * * (sudo python3 /home/pi/Desktop/ImagingSystem_AugSept2022.py)
#30 14 * * * (sudo python3 /home/pi/Desktop/ImagingSystem_AugSept2022.py)
#00 17 * * * (sudo python3 /home/pi/Desktop/ImagingSystem_JulyAug2022.py)
#00 16 * * * (sudo python3 /home/pi/Desktop/ImagingSystem_Oct2022.py)
#50 17 * * * /usr/bin/expect -f /home/pi/autom.exp
#00 16 * * * (sudo python3 /home/pi/Desktop/DkPlt_run_plan.py)
00 17 * * * (sudo python3 /home/pi/Desktop/ImagingSystemCode_CameraAonly.py)

[Read 48 lines]
^G Get Help  ^O Write Out  ^W Where Is  ^X Cut Text  ^J Justify  ^C Cur Pos  ^U Undo  ^M Mark Text  ^_ To Bracket  ^Q Previous
^X Exit      ^R Read File  ^N Replace   ^U Uncut Text ^T To Spell  ^G Go To Line ^E Redo    ^G Copy Text  ^Q Where Was ^W Next
```

Delete the hashtag “#” in front of the final line of code so as to make it an active task. The active codes becomes white so once you delete the hashtag at the beginning of the line, the line will

turn from blue to white. Once you made any change on the crontab, you should press Ctrl+X to save and exit. After pressing Ctrl+X, it'll ask you to save it. Press Y to save. Then, it'll ask you where to save the crontab. **Do not change the directory or file name.** Press Enter.

Note that this code is set to run at 17:00 (5pm) every day once you delete the hashtag. You can set the time by changing 00 and 17 in the code. If you want more than one arbitrary operations, you can simply copy and paste this line of code, and enter another time. You can also copy and paste the code and change the code file to have multiple imaging schedules running at the same time. If you want to periodically run any command/code/script, there are other available codes you can find over the net.

Once you have made your changes in crontab and save and exit using Ctrl+X, you will return to Pi Terminal. To double check your scheduled tasks, you can run crontab -l command to see your changed crontab.