OT-2 Robot
General Operations

Table of Contents

1. Main Robot Hardware

The Opentrons OT-2 robot is a liquid handling (pipetting) robot. It is made specifically for the life sciences, and is designed to mimic common lab pipetting routines. As such, the robot is suited to automate common pipetting operations. The robot operates in an exact manner, meaning that it do not have methods to sense its surrounding. It is thus important for its operators to perform the necessary hardware setup and maintenance for it to perform optimally. The following section will discuss about the robots' overall architecture and provide a general guide for the maintenance of its hardware.

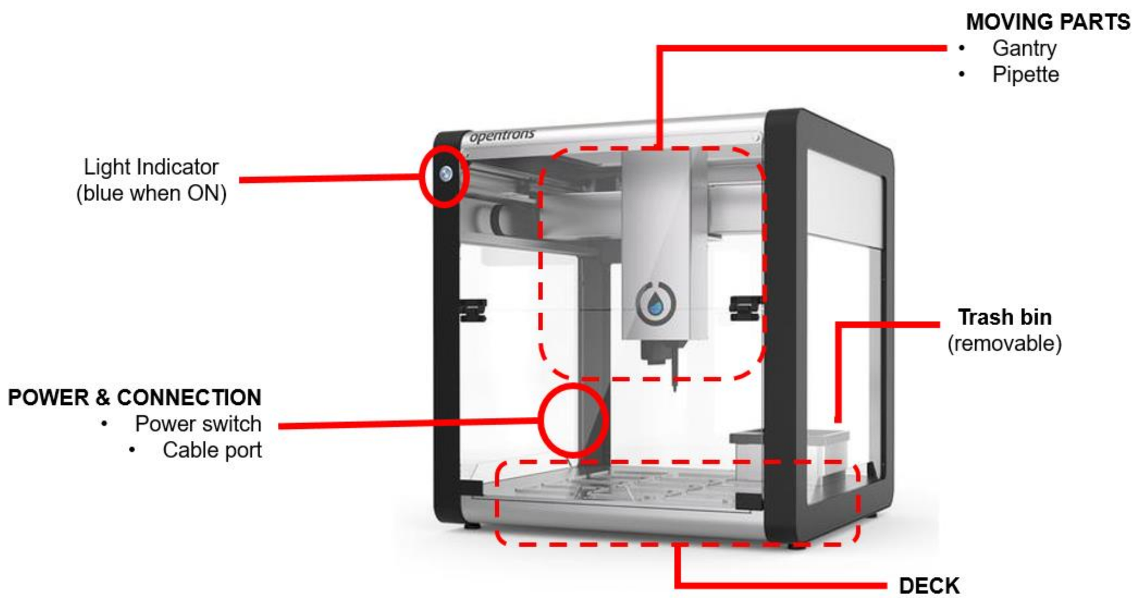

Fig. 1 | Robot main body

2. Modular Hardware

a. Pipettes

Every OT-2 robot has two pipette slots. These slots can be filled with any kind of OT-2 pipettes. So far, our lab has three types of pipettes, 1) P300 single-channel pipettes (GEN1 and GEN2), 2) P1000 single-channel pipette (GEN2), and 3) P300 multi-channel pipettes (GEN1 and GEN2). Although the pipettes can fit in both of the robot's pipette slots, it is very important to synchronize the pipette type and location with the protocol that you are using. When using the Opentrons protocol designer, you will be asked to declare which pipette will be placed in which pipette slot, and the software will tell you if a wrong pipette is attached when you load the protocol. <u>However, when using the Jupyter notebook, a similar warning will not be given.</u> Thus, the user have to make sure which pipette is being attached. Otherwise, users would have to declare this in the Jupyter notebook

The current default pipette location for the Jupyter notebook is:
Left : P300 MULTI-channel pipette (GEN1 for the 2018 robot, and GEN2 for the 2021 robot)
Right : P300 SINGLE-channel pipette (GEN1 for the 2018 robot, and GEN2 for the 2021 robot)

Pipettes can be changed using the Opentrons application. This operation will be described at a later section.

b. Deck and Labware

The robot deck provides 11 slots in which labwares (i.e. tube racks, well plates, tipracks, etc…) will be placed during the run. The deck slots are numbered in the actual robot. The numbering configuration is shown on Fig. 2 (slot 1 is located on the left side and is on the row closest to the door). The eleven slots are of equal dimensions, meaning that any type of OT-2 compatible labwares can be placed in any slots without restriction. However, the actual labware would have to be declared in the run protocol. If you are using Opentrons' protocol designer, you would be asked to declare where the labwares are to be located. Alternatively, if you are using the Jupyter notebook, this declaration has to be performed in the Python script. However, when the web server is used, the servers' output will inform you where the labwares should be placed on the deck.

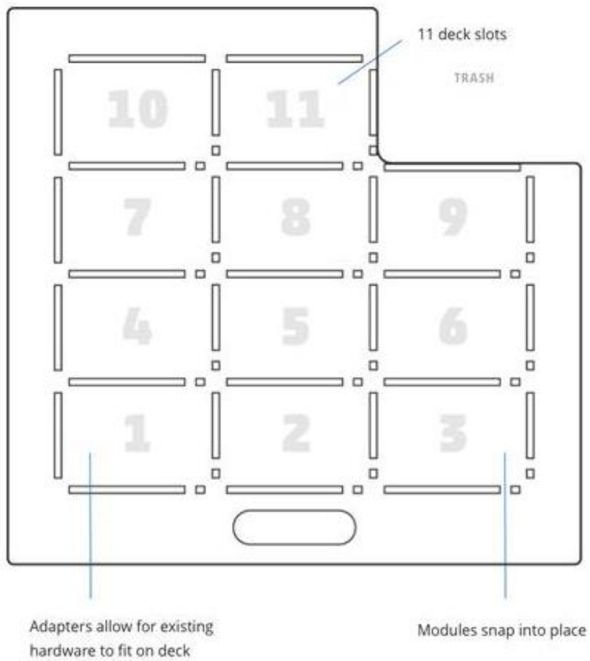

Fig. 2 | Robot deck overview

During a run, the deck slots are to be filled with the required labwares. In general, there are three kinds of labwares, 1) tube racks, 2) tip racks, and 3) well plates.

Tube Rack

The tube rack consists of the stand and the interchangeable tube holders (Fig. 3A). The bottom of the stand is generic and should fit on the deck's slots (Fig.3B). The interchangeable tube holders can be placed on top of the stand to hold different types of tubes (i.e. 1.5 and 2.0 mL Eppendorf tubes, as well as 15 and 50 mL Falcon tubes). The tube holder and the stand can be aligned by using the diagonally cut end of the tube holder (Fig. 3C). Once aligned, the two can be combined easily and should give a light "click" sound when attached.

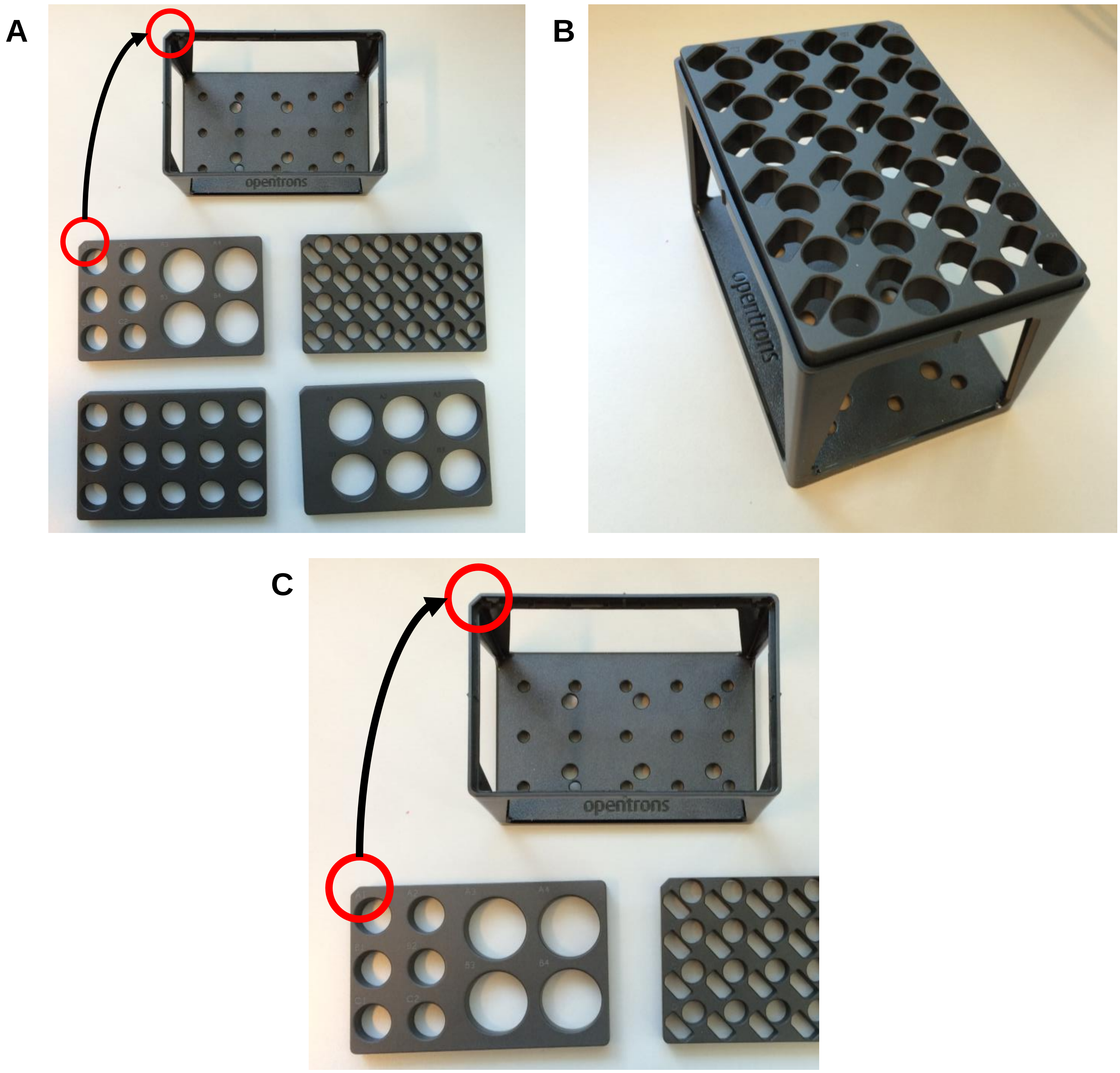

Fig. 3 | Modular tube racks

Tip Rack

Currently, the OT-2 robot can only be used with the Opentrons tip racks (self-made tip racks may also be used, but additional protocol would be needed). The tips itself are compatible with usual Gibson micropipettes, meaning that the robot might be compatible with common P300 pipettes. Nevertheless, whenever possible, please use the tips and tip racks provided by Opentrons. Unless scripted to do so, the robot will always begin the protocol by using the top left tip on the tip rack. Thus, it is important to make sure the top-left section of the rack is filled when using a half-used tip box.

It is important to open the tip racks' lids before each run. The robot runs on exact operation, meaning that it would attempt its usual operation and stab the rack lid if you forget to remove them.

Well Plates

So far, OT-2 robot is compatible with BioRad, Corning, and NEST plates. The up-to-date list of compatible well plates can be seen here: https://labware.opentrons.com/?category=wellPlate. In general, well-plates made by other makers (i.e. Greiner) can also be used in place of these specific well-plates even though a labware recalibration would be required before using a new type of well-plate. For example, Greiner 96-well plates (the one that we currently use in the lab) is not in the list, but can be used as a BioRad 200 µL 96-well plate after adjusting the labware calibration.

3. Hardware Clean-Up

While used for growth experiments, it is necessary to keep the inside of the OT-2 robot sterile. This can be performed by wiping the deck, sides, and the pipettes with 70% isopropanol and tissue papers. Do not use ethanol. The pipettes can be difficult to reach. Use the protocol: "PippeteWipe.pynb" (located on the Jupyter Notebook home directory) to move the pipettes to the front one by one. Make sure the deck is completely empty before running this script. The script will prompt your confirmation before every move.

4. Communications with the Robot

The robot can be communicated through two different applications, namely the Opentrons application and the Jupyter notebook. The Opentrons application is an essential mode of communication with the robot as this is where the user would perform most of the setup and basic operations. The Jupyter notebook is used only for advanced and complex operations, such as the scripted operations.

a. Opentrons Application

The Opentrons application is the main communication interface with the robot. In our lab PC, the application can be found on the desktop (Fig. 2). When using a new PC, the Opentrons application can be freely downloaded from the following link.

https://opentrons.com/ot-app/

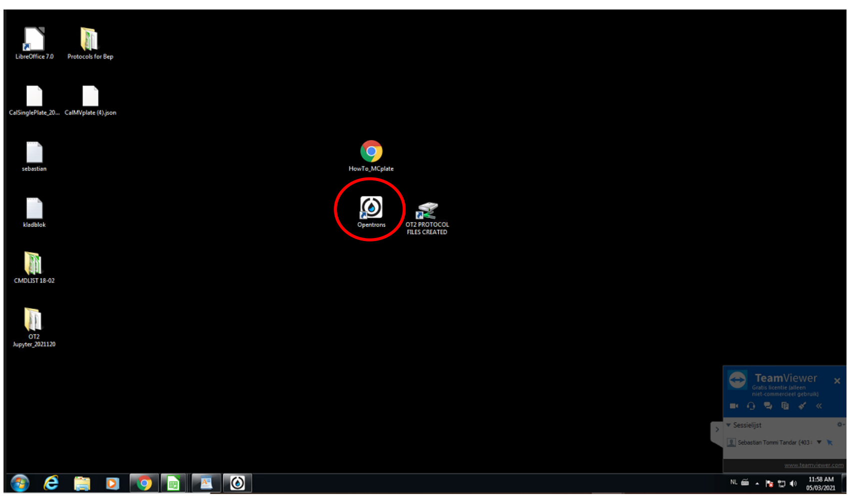

Fig. 4 | Opentrons application on lab PC

Wired connection

The Opentrons application is currently the only reliable application through which you can connect to the robot. When using a cable connection, connected robots will be shown on the top-left side of the application. Robots will only show up in the application if the robot is turned ON (this may take 3-5 minutes after the power switch is turned ON). To connect to the robot, simply click on the toggle switch beside the robot's name (Fig. 5). This connection has to be made in the beginning of every run.

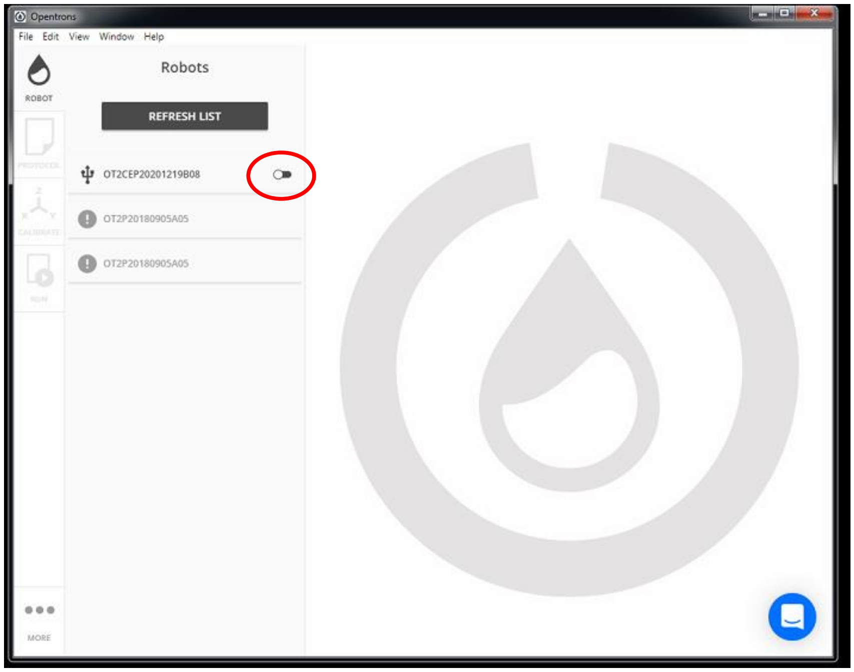

Fig 5 | Connecting to the robot

Wireless connection

The robot can also be connected to a wireless network. To do this, you would have to first connect with the robot using a wired connection. After the making the wired connection, scroll down to the "Connectivity" section. On the drop-down menu, you can select which network to connect the robot to (Fig. 6). The application will ask you to enter a password if required. The robot will automatically re-connect to the network whenever it is turned ON. In other words, the wireless connection setup would only need to be done once.

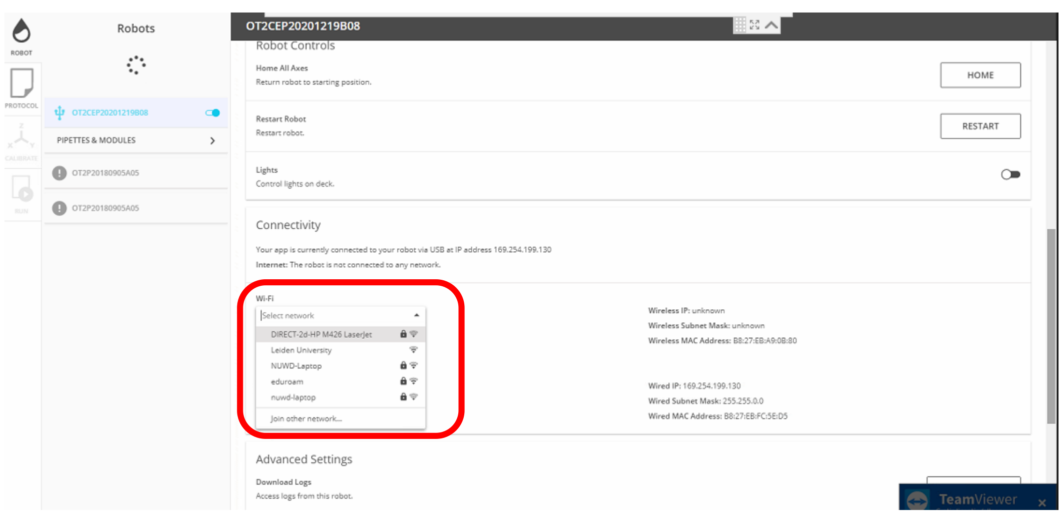

Fig 6 | Connecting to the robot to a wireless network

Once connected to a network, any PC connected to the same network can control the robot using an Opentrons application (Fig. 5). It is important to note that not all wireless networks can be used for robot connection (the eduroam wifi, for example, cannot be used).

Both robots are currently connected to the ad-hoc network made from incubator robot (host) PC. To start the ad-hoc network, open command prompt on the host PC. The command prompt has to be opened as administrator. Type the following command to start the wireless network connection (Fig. 7A):

netsh wlan start hostednetwork

To stop the network, type the following to the command prompt (Fig. 7B):

netsh wlan stop hostednetwork

To reconfigure the network with a new name and password, type in the following commands (Fig. 7C). However, you would have to reset the connection to the robots with a wired connection (Fig. 6).

***netsh wlan set hostednetwork mode=allow ssid=*(chosen network name; no space) key=(password)**

Any PC that can connect to a wireless connection can start this type of ad-hoc connection (including your own laptop!).

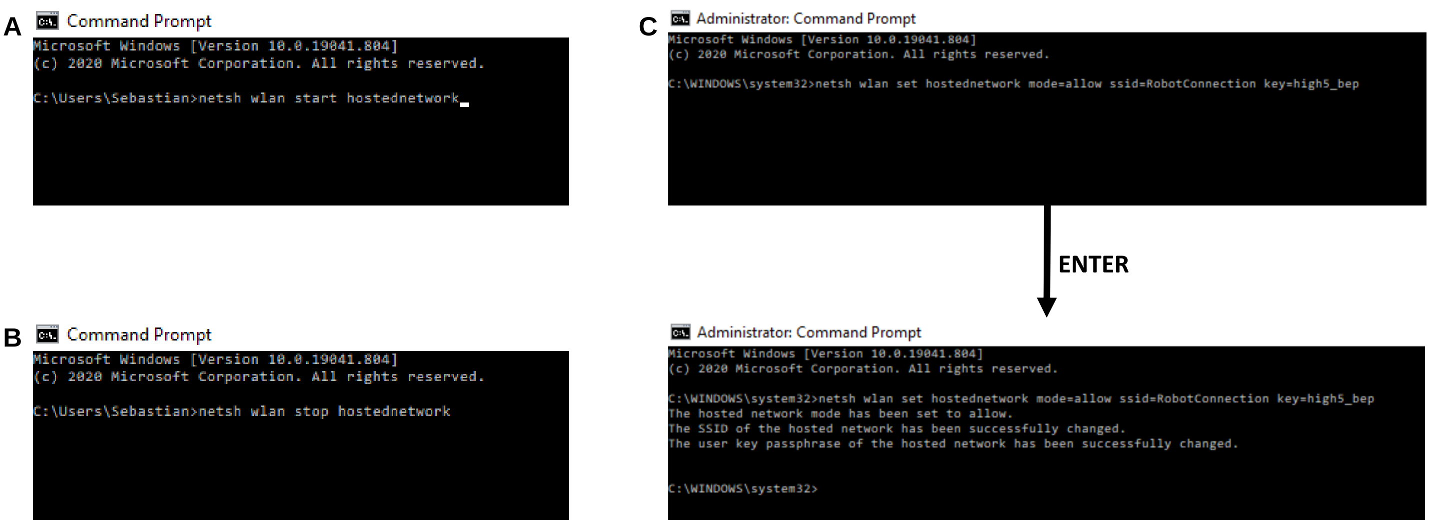

Fig. 7 | Ad-hoc connection setup in command prompt

Homing command

The **Home** button (Fig. 8) is important to reset the robot's gantry to its original location (top, back, right side of the robot). This home location is the safest position of the robot. The homing path is also made in such way that it would not hit any items located on the robot deck, regardless of the gantry/pipette's current location. The **Home** command is also important to reset the robot's location memory. It is thus important to home the robot before each use (separate command is also available on the Jupyter notebook for scripted homing).

This button is extremely important when a protocol run is stopped before the end of the run. In such cases, before doing any other operations, press the **Home** button to reset the robot location and ensure the machine is located at a safe location.

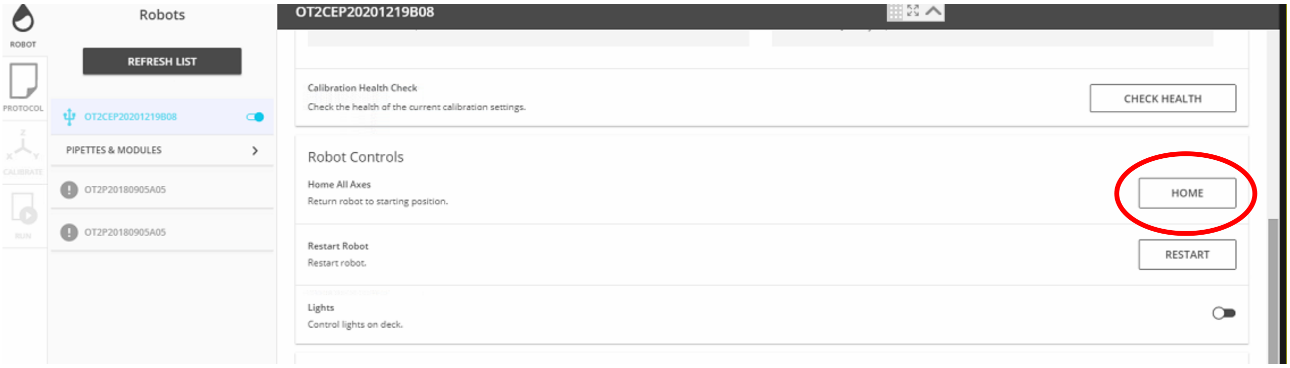

Fig. 8 | Home button

Creating and running a ".json" protocol

In general, there are two ways to create and execute an OT-2 protocol, 1) using the Opentrons protocol designer, and 2) by using the Jupyter notebook. Here, we will discuss about the first mode of operation. The Opentrons protocol designer can be accessed online (https://designer.opentrons.com/). To this moment, the protocol designer is still on its beta version. As such, even though it can be used to perform simpler operations, it is not yet optimized to perform complex operations (i.e. adjust pipette height according to liquid height).

Step 1. Open the Opentrons protocol designer
Step 2. Create the protocol. The web page will guide you through the necessary steps
Step 3. After you finish the protocol design, click **FILE**, then click **EXPORT.** This will download your protocol as a ".json" file
Step 4. Turn ON the OT-2 robot, open the Opentrons application, and connect to the robot
Step 5. Click on PROTOCOL (Fig. 9, red circle)
Step 6. Browse OR drag and drop the downloaded ".json" protocol to the given location (Fig. 9). The loading will take a couple of minutes.

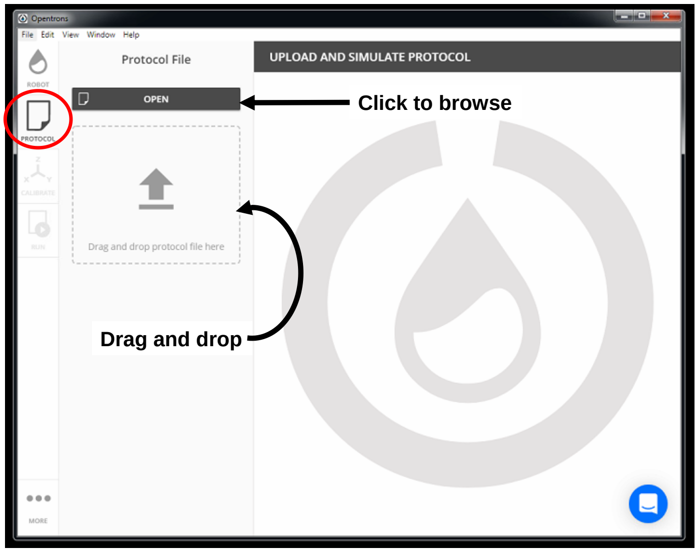

Fig. 9 | Uploading .json protocols

Step 7. Continue with the labware calibration. The application will guide you through the steps

- Sometimes, pipette calibration will also be needed. The application will inform you if this is the case.
- The application will give a warning if the wrong type of pipette is connected.
- This calibration can sometimes be skipped. However, just to be save, do the calibration if this is first time the current ".json" protocol is being loaded. The application will prevent you from continuing further if the calibration is necessary

Step 8. Click **RUN** to begin the protocol

b. **Jupyter Notebook**

The Jupyter Notebook is used to perform custom scripted operations. However, it cannot be used to perform basic operations such as connecting to the robot or calibration. The notebook can be accessed through the Opentrons application (Fig. 8). After clicking the **Open Jupyter Notebook** button, a Google Chrome window will be opened, and you will be directed to the robot's main directory.

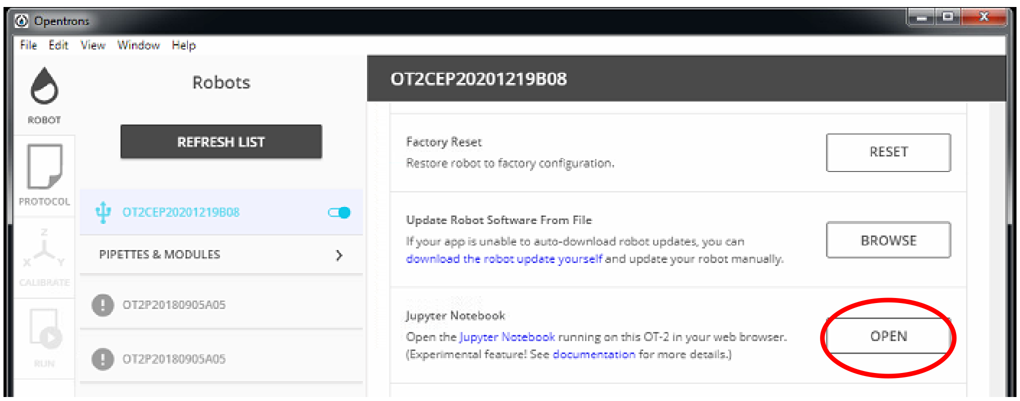

Fig. 10 | Opening the Jupyter notebook

The Jupyter Notebook can be operated as common Jupyter Notebook. However, the current software prevents user from installing non-default python libraries to the OT-2 Jupyter Notebook (default libraries like 'math' are available, non-default libraries like 'pandas' are not available).

Jupyter Notebook directories

The accessible directories in the robot's Jupyter Notebook is quite thin. Currently, the directories are composed of the following items (Fig. 11).

/home

- **Archive** : storage for old versions of the script
- **User Inputs** : directory to input the robot commands (.csv files) obtained from the web server
- **OT2_MainController.ipynb** : main controller (shortest, calls "MainExec.py")
- **PipetteWipe.ipynb** : pipette wipe helper
- **OT2_ControllerLib.py** : main function library (edited during updates)
- **MainExec.py** : protocol executor; main shell called in the main controller

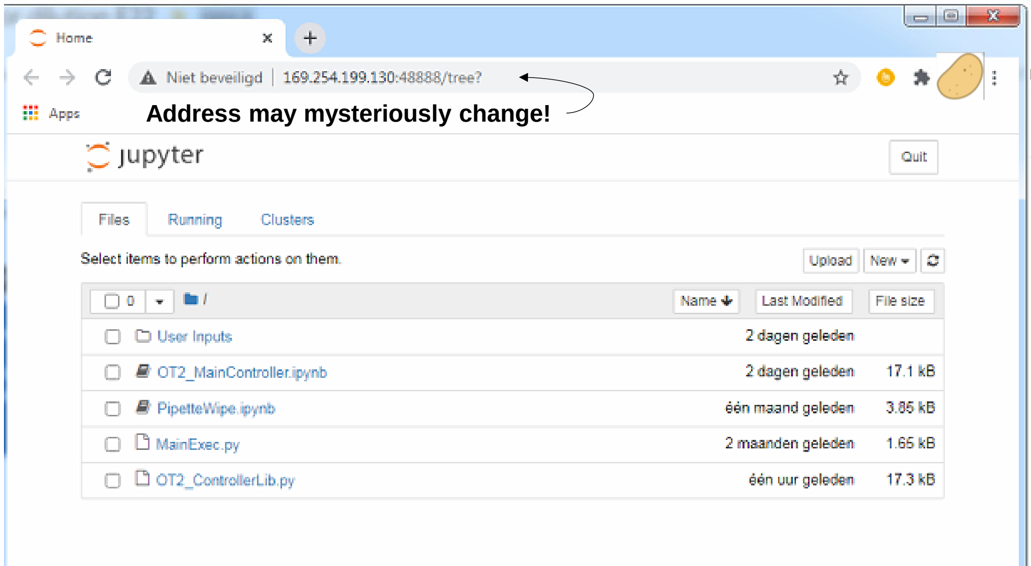

Fig. 11 | Home Jupyter notebook

Writing custom Jupyter scripts

The robot's operation can be scripted using Python. The overall guideline is accessible from the following web page.

https://docs.opentrons.com/v2/new_advanced_running.html

To execute the Opentrons command script in your own PC, you would first need to install the 'opentrons' package to your local Python library (https://pypi.org/project/opentrons/). For troubleshooting, it is also possible to simulate a run without running the robot using *opentrons.simulate* commands.

Running a protocol using Jupyter Notebook

Using the general script:

Step 1. Turn ON the robot, open the Opentrons application, connect to the robot, and open the robot's Jupyter Notebook.

Step 2. Open the UserInputs directory.

Step 3. Upload your .csv command list (can be downloaded from the ShinyApp web server).

  a. Click "Upload" (Fig. 12A)
  b. Select your .csv file
  c. Click "Upload" (Fig. 12B)
  d. Tips: you would need the complete file name of your command list for step 8

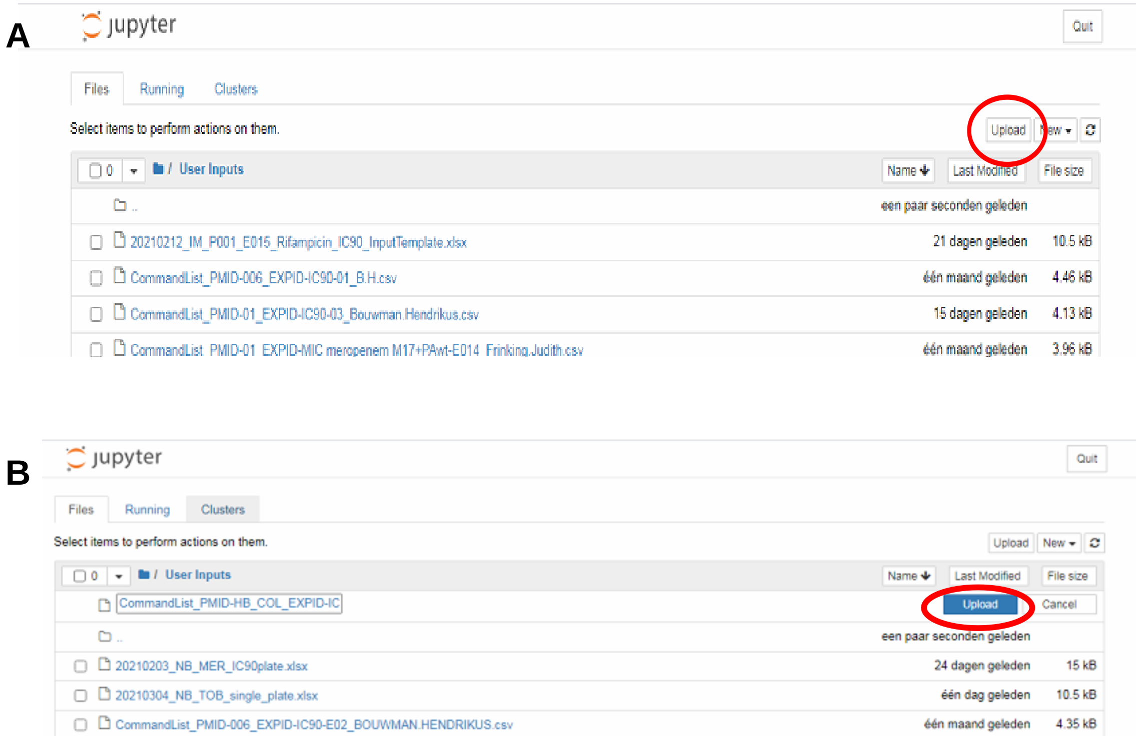

Fig. 12 | File upload buttons in Jupyter Notebook

Step 4. Go back to the **home** directory by clicking on the folder icon (Fig. 13)

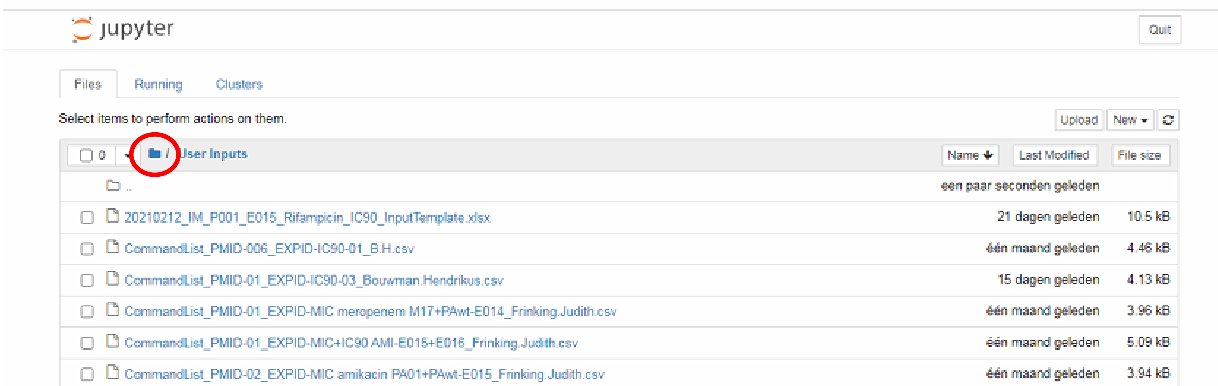

Fig. 13 | Shortcut to home directory in Jupyter Notebook

Step 5. Open **OT2_MainController.ipynb** (see Fig. 11)

Step 6. Click the **Refresh Kernel** button (Fig. 14)

Step 7. Click the **Run** button (Fig. 14)

Step 8. Paste the file name of your command list (uploaded on step 3) to the input tab (Fig. 14), then press ENTER

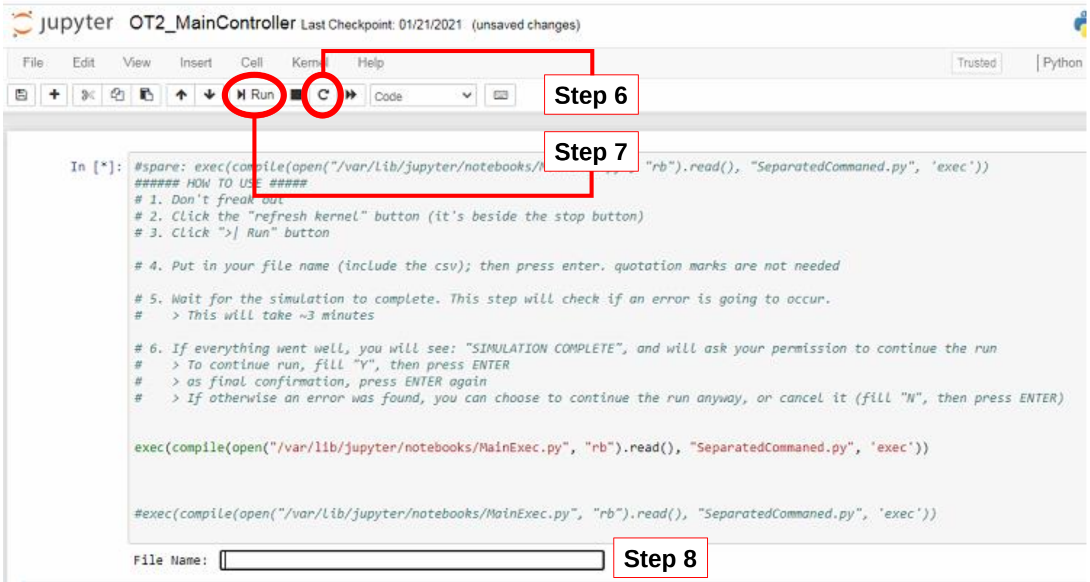

Fig. 14 | Running the main controller

Step 9. Fill "Y" if you want to do a simulation run (check if anything would go wrong in the middle of the run), or "N" to skip the simulation, then press ENTER

Step 10. After the simulation (3~5 minutes), fill "Y" to continue with the actual run, or "N" otherwise, then press enter

Step 11. Make sure all the deck preparation is ready before continue

Step 12. Press enter to actually start the run

In case of emergency, abort the run with the stop button (Fig. 15).

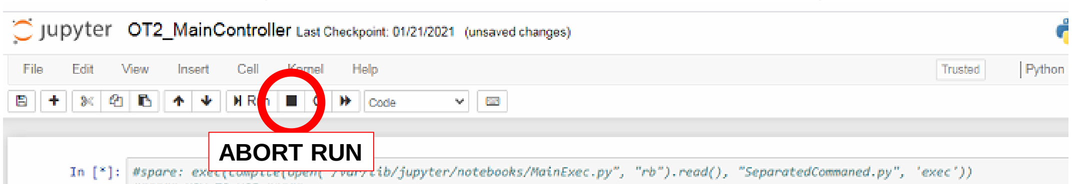

Fig. 15 | Interrupt run button

Using a custom script:

Step 1. Turn ON the robot, open the Opentrons application, connect to the robot, and open the robot's Jupyter Notebook.

Step 2. Upload your .pynb script to the robot's directory

Step 3. Open the script by clicking on the item.

Step 4. Click the **Refresh Kernel** button (Fig. 14, button indicated as "step 6")

Step 5. Click the **Run** button (Fig. 14, button indicated as "step 7")

Step 6. The run will start immediately

4. **Pipette, Deck, and Labware Calibrations**

Calibrations is essential for the robot to correctly aim the pipette at a specific location on the deck. Currently, the types of calibration needed continues to be updated by Opentrons. Nevertheless, the Opentrons application provides a step-by-step walkthrough for each processes. In the following section, we will go through the currently essential types of calibration, and how to access the calibration walkthrough on the Opentrons application.

Deck calibration

Deck calibration has to be performed every time the robot is moved. Before starting the deck calibration, it is necessary to check the level/flatness of the deck. To do this, use the bubble level/spirit level (Fig. 8, top). Place the bubble level in the middle of the deck. Adjust the robot's feet until the bubble level is exactly at the center of the apparatus. To do this, you would need an OT-2-compatible wrench (Fig. 8, bottom).

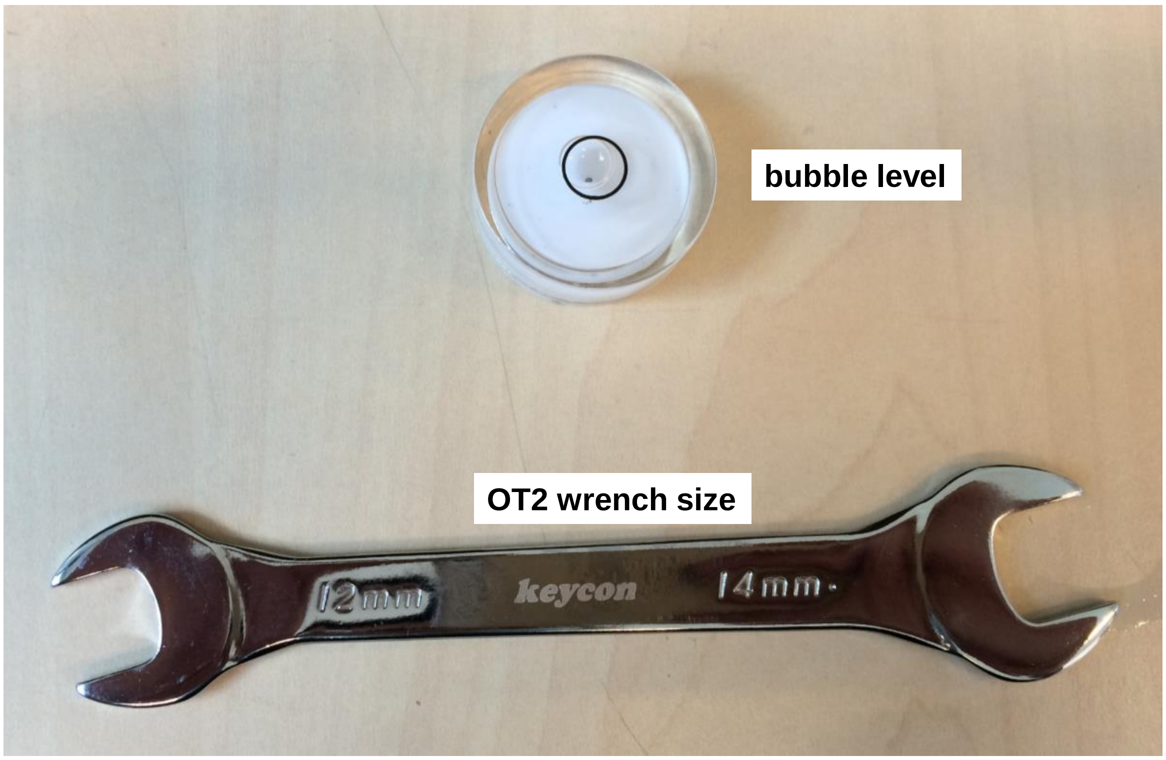

Fig. 16 | OT2 calibration tools

To start the deck calibration, open the Opentrons application and connect to the robot. Scroll down to find the "Recalibrate Deck" button (Fig. xx, top marker). Click the button, and acknowledge the warnings. Make sure the deck is completely empty for following operations. The application will guide you through the calibration process.

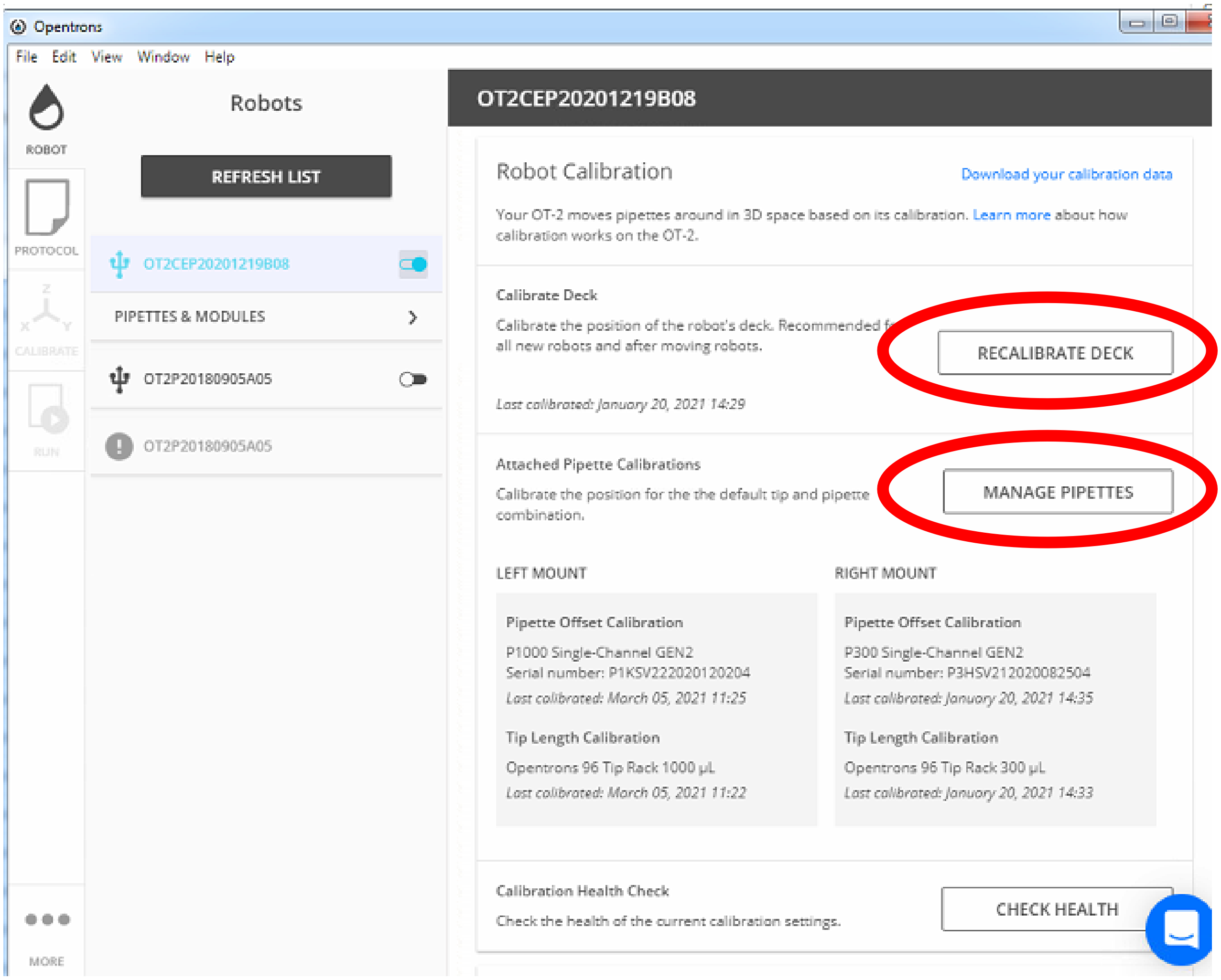

Fig. 17 | Calibration shortcuts

Pipette calibration

Pipette calibration has to be performed every time a new pipette is re-attached.

Step 1. Click “Manage Pipettes” (Fig. 17, bottom marker)
Step 2. Select one of the calibration options, then follow the application’s guide (Fig. 18, arrow markers)
Step 3. Follow the application’s guide

Sometimes, you are going to need the calibration block to calibrate the pipette (Fig. 19).

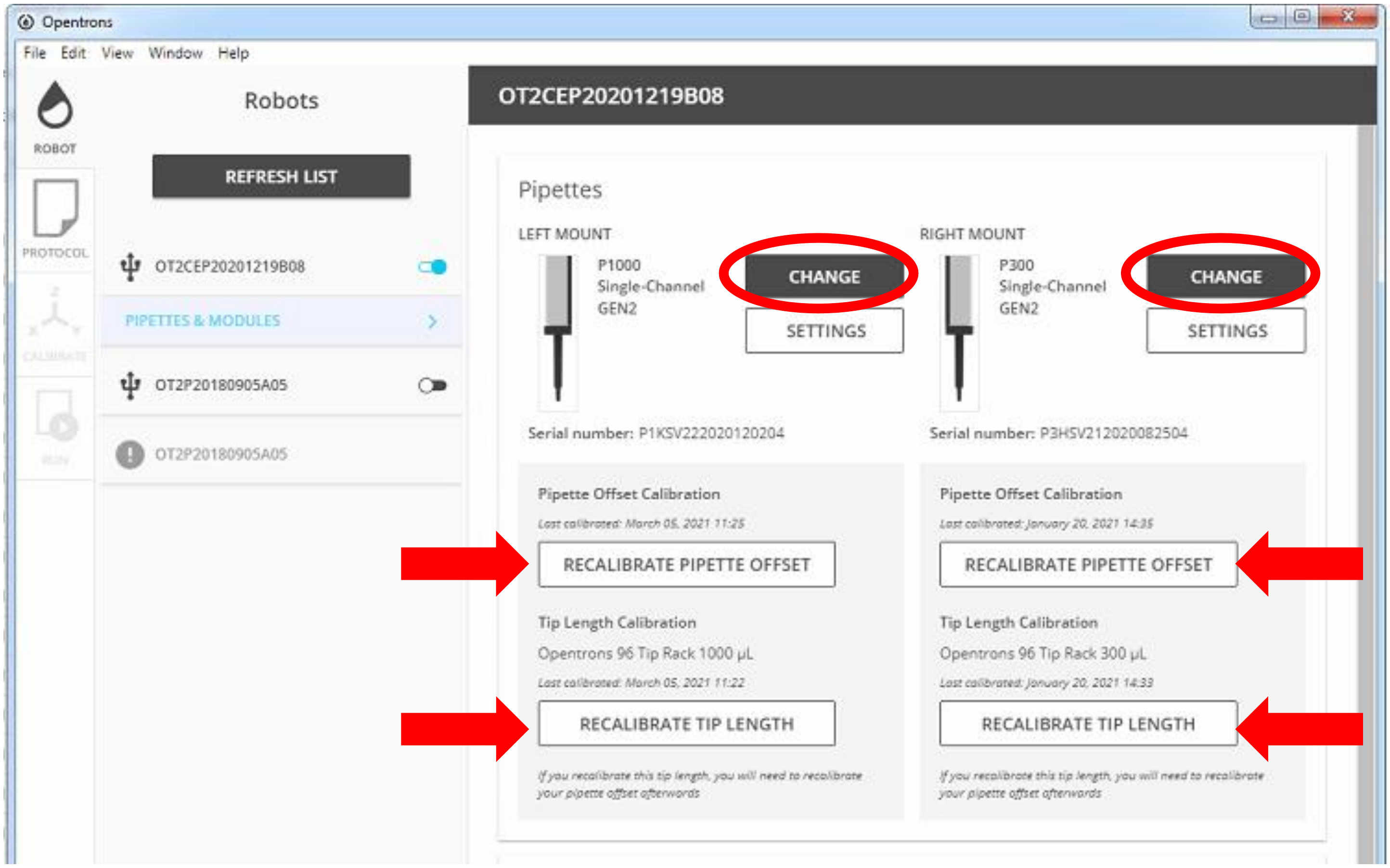

Fig. 18 | Manage pipette menu

Fig. 19 | Opentrons Pipette Calibration Block

Labware calibration

So far, labware calibration can only be performed using by loading a ".json" protocol on the Opentrons Application. Labware calibration will automatically be prompted when you load a ".json" protocol. However, when using a custom Jupyter Notebook protocols, it might be good to mimic the labware calibration as the following.

Step 1. Note down the labwares and the locations in which they will be placed at during the actual run using the Jupyter Notebook protocol.
Step 2. Go to the Opentrons protocol designer web page (https://designer.opentrons.com/)
Step 3. Create a new protocol, and make sure the correct types of pipettes are loaded
Step 4. Place labwares according to you Jupyter protocol (noted on step 1)
Step 5. Place tubes/liquid at slot A1 on each labwares
Step 6. Create a dummy run such that each of the labwares' slot A1 is used at least once during the run.
Step 7. Export the protocol
Step 8. Turn ON the robot, open the Opentrons application, and connect to the robot
Step 9. See: ***Creating and running a ".json" protocol*** on how to load and perform the labware calibration. The dummy protocol does not need to be executed to complete the calibration.

Changing pipettes

Pipettes are part of the modular hardware. This operation can be quite complex, but the Opentrons application will provide a step-by-step guideline (with pictures) for the process. To do this, you would need the 2.5 mm screwdriver (Fig. 20).

Fig. 20 | 2.5 mm screwdriver for pipette changing

Step 1. Turn on your OT2 robot, and open the Opentrons application on the PC
Step 2. Make sure the robot deck is completely empty
Step 3. After connecting to the robot, click "Manage Pipettes" (Fig. 17, bottom marker)
Step 4. Click "Change Pipette" (Fig. 18, oval markers)
Step 5. **Follow the guide on the application.** This guide is kept up-to-date by Opentrons and becoming increasingly clear.

<u>Tips:</u> It is important to firmly attach the pipettes' cables. This can be a bit tricky to do. Once the pipette is moved to the front, it is possible to gently pull down the pipette to expose the cables. After you connect the cables, you can gently push the pipette back to its original height before continuing the rest of the procedure.

Factory reset

The factory reset button allows user to erase all calibration memory from the robot. After initiating the factory reset, the robot will have to be re-calibrated completely before use. This function is useful in rare cases where the pipette's z-axis memory is accidentally set to below the deck level (can happen when calibration is interrupted, either by command or power/connection disruptions).

5. Supporting Web Server

The ShinyApp-web server was made to translate plate maps into OT-2 commands. Currently, there are four types of operation-oriented applications and one measurement pre-processor application. The main directory of the server can be accessed through http://132.229.100.197:2222/ot2/. The different applications can be accessed from this main directory using their given name.

[1] Single Plate Processor
Accessible from : http://132.229.100.197:2222/ot2/SingleplateMIC/
Accessible from main directory as : SingleplateMIC
Processor for filling ONE 96-well plate with drug solutions. The processor can only process a maximum of ONE DRUG PER-WELL, even though there is a less stringent limit to how many drugs can be included per-plate (this depend more on the tube rack availability).

[2] Multiple Plate Processor
Accessible from : http://132.229.100.197:2222/ot2/MVPlate/
Accessible from main directory as : MVPlate
Processor for filling MULTIPLE 96-well plate with drug solutions. The processor can only process a maximum of ONE DRUG PER-WELL, even though there is a less stringent limit to how many drugs can be included per-plate (this depend more on the tube rack availability).

[3] Drug Combination Processor (Multiple Plate)
Accessible from : http://132.229.100.197:2222/ot2/CQ_Plate/
Accessible from main directory as : CQ_Plate
Processor for filling MULTIPLE 96-well plate with drug solutions. The processor can process a maximum of MULTIPLE DRUGS PER-WELL. However, the complex dilution performed in this operation may significantly increase robot run time.

[4] Custom Medium Mixer
Accessible from : http://132.229.100.197:2222/ot2/M9MixR/
Accessible from main directory as : M9MixR
Processor for mixing custom medium solutions. Each medium may have a maximum of 50 mL size. The effective limit of the number of components depend on tube rack availability.

[5] Growth Curve Pre-processor
Accessible from : http://132.229.100.197:2222/ot2/PlateAnalysis/GrowthCurve/
Accessible from main directory as : PlateAnalysis > GrowthCurve
Takes plate map and spectrophotometer readout file(s) as input. This pre-processor combines measurement data from all measurement files and combine them into a single long-format file. Actual measurement time (as noted by the machine, not by the file name) will be used for tagging time. Measurement values from wells with identical drug-medium condition will be combined as a single measurement based on its average. Preliminary plots were made using the pre-processed data.