Manual for the OT-2 protocol selector program

This document contains instructions on how to use the OT-2 Protocol Selector program, as well as a troubleshooting section with solutions and more information on some issues that might occur while using the robot. The manual also contains information on which procedures require using the official Opentrons app, as well as how to perform these. The final section contains a brief description of the general structure of the code meant for advanced user that may want to make modifications to the program.

The program and this accompanying manual were made during the Design-Build-Test 2021 course by:

Agata Jasna, Elsa Renström, Johan Lehto, Johan Lundström, Mathias Jonsson and Tiam Fitoon.

Table of contents

Using the OT-2 protocol selector program

The OT-2 Protocol Selector program allows for creating and running protocol on the Opentrons OT-2 robot. The program supports SPRI beads DNA cleaning protocols and qPCR preparation protocols, which are constructed based on parameters input by the user or by a .csv file provided by the user, respectively.

The general workflow when using the program is as follows:

1. Turn on the robot using the power switch.
2. Wait for the light on the front of the robot to shine with a steady blue light before plugging in the USB cable to a computer.
3. Start the OT-2 Protocol Selector program.
4. Select the type of protocol that you want to work with.
    a. For SPRI bead DNA cleaning: Enter the desired protocol parameters and press “Create robot protocol”.
    b. For qPCR preparation: Press “Choose a file” and select the .csv file to base the protocol on.
5. (Optional: Press “Estimate time” to simulate the protocol using an experimental function from the robot manufacturer to get a rough estimation on how long the robot will run for.)
6. Press “Next”. A new window will open showing instructions on the necessary steps needed to prepare the robot for running the protocol.
7. Check the connection to the robot by pressing the “Check connection” button. If the connection fails, try pressing the button a few more times, otherwise see the troubleshooting section for more information.
8. Following the instructions and load the robot deck according to the picture. When running a qPCR protocol, also load each tube rack according to its own tab.
9. (Optional, for qPCR protocol only: Press “Print layout to file” to get a printable version of the tube rack layouts and the volumes needed. This will create a .txt file in the same folder as the provided .csv file, which then can be printed by the user if needed.)
10. Once ready, press “Run protocol” to start the robot. The robot will output information about each step it is doing in the terminal window that also opens when starting the program. The robot can be paused by opening the front door.
11. When the protocol is finished, closed the program by pressing the “Exit” button. Once the program has closed, the robot can be shut off using the power switch and the USB cable can be unplugged.

Using the Opentrons app

Some procedures cannot be done using our program and instead require the official Opentrons app, which can be downloaded from https://opentrons.com/ot-app/. This includes changing pipettes and calibrating the robot.

Note that it if our "OT-2 Protocol Selector" program has been used to the extent that a protocol has been created and the deck layout has been shown (i.e. the "Next" button has been pressed after entering protocol values for a SPRI bead protocol or after a .csv has been chosen for a qPCR protocol), it is important to properly close the program using the "Exit" button. Otherwise, the computer may have problems connecting to the robot using the official app and the robot must be restarted.

Changing pipettes

To change the pipettes, the official Opentrons app must be used. If the app has problems connecting to the robot, see the troubleshooting section of this manual.

The following is a step-by-step instruction for changing pipettes. If the robot is already running and connected, skip directly to step 4.

1. Start the robot.
2. Wait until the light on the front shines with a steady blue light.
3. Connect to the robot by USB.
4. Open the official Opentrons app and wait for it to identify the robot.
5. Once a connection has been found, enable it using the toggle switch by the robot name.
6. Select change pipettes and follow the instructions in the app.

Calibrating the robot

Our protocols have been designed and tested to not require robot calibration each time they are ran. Still, it is recommended to occasionally calibrate the pipettes and labware to ensure optimal pipetting. This must also be done using the official Opentrons app.

The following is a step-by-step instruction for calibrating the robot. If the robot is already running and connected, skip directly to step 4.

1. Start the robot.
2. Wait until the light on the front shines with a steady blue light.
3. Connect to the robot by USB.
4. Open the official Opentrons app and wait for it to identify the robot.
5. Once a connection has been found, enable it using the toggle switch by the robot name.
6. Select the calibration tab on the left side and follow the instructions in the app.

Manually editing the code

Certain parts of the program and protocols are hard coded to use specific labware or might accidentally be hard coded for aspects that are specific to the robot used. If different labware is to be used in the future, or if the program is to be used on another robot these might potentially have to be edited. Here follows instructions on how to change some of the hard coded aspects.

Changing pipette model

The protocols are hard coded to use Gen 1 P10 and Gen 1 P300 pipettes and will not run if the robot does not detect the specified model. If wanting to use different pipettes:

1. Open the blueprint file for the protocol that is to be edited.
   1.1. "dna_cleaning_blueprint.py" AND "dna_cleaning_blueprint_few_samples.py" for the SPRI beads protocol.
   "qpcr_blueprint.py" for the qPCR protocol.
2. Before editing, find the exact name of the new pipette model from the Opentrons website https://docs.opentrons.com/v2/new_pipette.html#pipette-models
3. Edit the model name to the name of the new pipette
   The pipettes are initialized at the start of the `run()` function by
   `protocol.load_instrument(model, mount).`
   For example, changing a Gen 1 P10 single channel pipette mounted on the left side to a Gen 2 P20 single channel pipette, one would change
   `protocol.load_instrument('p10_single', 'left')` to
   `protocol.load_instrument('p20_single_gen2, 'left')`
4. Note: It is important to not change the variable name (p10 = protocol.load_instrument()) even if the pipette is changed to a non-P10 model.

Changing pipette tip model

The program is hard coded to use Opentrons' 20 µl and 300 µl pipette tips. However, different manufacturers might have slightly different dimensions on their tips or on the tip box and if the different tips are to be used the code must be changed to reflect this. Note that the robot has also been tested using GEB 10 µl and 300 µl pipette tips, which are interchangeable with the 20 µl and 300 µl Opentrons tips, respectively, and thus not require any modification to the code.

Some pipette tips are not available as pre-made labware and must be measured and created as custom labware. See the section on custom labware for instructions on that.

1. Open the blueprint file for the protocol that is to be edited.
   1.1. "dna_cleaning_blueprint.py" AND "dna_cleaning_blueprint_few_samples.py" for the SPRI beads protocol.
   "qpcr_blueprint.py" for the qPCR protocol.
2. Before editing, find the exact name of the new pipette tip model from the Opentrons labware library: https://labware.opentrons.com/?category=tipRack
3. Edit the model name to the name of the new pipette tips.
   The tips are initialized at the start of the `run()` function by:
   `protocol.load_labware(model, location)`

Custom labware

Not all pipette tips are available as pre-made labware for the robot to use and must thus be added as custom labware, which require a bit more work. Other types of custom labware can also be added such as custom-made tip racks.

1. Create the custom labware using the labware creator on the Opentrons website: https://labware.opentrons.com/create
2. Control the newly created custom labware by following the Labware Test Guide, found on https://opentrons-publications.s3.us-east-2.amazonaws.com/labwareDefinition_testGuide.pdf
3. After following the guide, the custom labware will only be available when running protocol using the official Opentrons app. To make the labware available when running protocol over SSH (such as when using the "OT-2 Protocol Selector" program) continue with the following steps.
4. Upload the labware definition by running the following command from the command prompt while connected to the robot:
   `scp -i C:\Users\LOCALUSER\Opentrons\ot2-ssh-key FILEPATH root@ROBOTIP:/data/user_storage/FILENAME`
   Where LOCALUSER is the name of the logged in user.
   FILEPATH is the full filepath to the newly created .json file that contains the custom labware definition. For example C:\Opentrons\custom_labware.json
   ROBOTIP is the wired IP shown in the official Opentrons app.
   FILENAME is the name of the .json file that contains the custom labware definition.
5. Edit the protocol blueprint file that should use the newly created labware by adding the following to the start of the run() function:

```
import json
path_to_custom = '/data/user_storage/FILENAME
    if protocol.is_simulating():
        path_to_custom = LOCALFILEPATH\\FILENAME
    with open(path_to_custom) as labware_file:
        labware_def = json.load(labware_file)

    labware1 = protocol.load_labware_from_definition(labware_def, LOCATION)
```

   Where FILENAME is the name of the .json file that contains the custom labware definition. LOCALFILEPATH is the full path to the folder where the .json file is stored on the local computer (only required to be able to simulate and estimate run time of the protocol). LOCATION is the deck slot in the robot where the labware will be placed.

Troubleshooting

The following describes some errors we found while working on the robot and how to solve them.

Connection errors

The program returns "Connection failed" when attempting to check the SSH-connection.

Solution:

1. Attempt to connect a few times. The program automatically changes between addresses it attempts to connect to so if one fails, another will the checked when the button is pressed again.
2. Restart the robot a wait a few minutes after the front light shines with a steady light before connecting the USB cable.
3. Connect to the robot using the official Opentrons app and then attempt to the SSH-connection again.
4. If still not able to connect over SSH, a non-recommended, emergency solution is to edit the IP directly in the python program.
    4.1. Connect to the robot using the official Opentrons app and check the wired robot IP.
    4.2. Open the "main.py" file and manually change the IP the program uses. This is done by editing the ip1 variable on line 37.

Issue: Unable to connect to the robot using the official app.

Solution:

1. Ensure that the robot has fully started before attempting to connect. First turn on the robot using the power switch, then wait for the light at the front to shine with a steady blue light until finally connecting the USB cable.
2. Try to connect several times using the "Try again" button. Sometimes the app is unable to establish a connection on its first attempt.
3. Restart the robot.
4. Ensure that the "OT-2 Protocol Selector" program has been closed properly using the dedicated "Exit" button if it has also been used to check the SSH connection.

More information: When "OT-2 Protocol Selector" program shuts down the opentrons-robot-server used for GPIO connections when a SSH connection has been successfully made. This is required for the program to be able to sense if the door is open and thus for it to be able to pause on door opening, as well as for the program to be able to control the lights in the robot. The opentrons-robot-server is then restarted when closing the program with the "Exit" button, which is why it may take up to 1 minute for the program to shut down. The official app uses the opentrons-robot-server to some capacity and might thus have problems connecting if the opentrons-robot-server is not properly restarted.

Calibration problems

Issue: The robot does not aspirate or dispense the same amount of liquid every time.
Issue: The robot is not centered when picking up pipette tips or transferring liquid.

Solution: Re-calibrate the robot. See the section on calibration for a step-by-step instruction.

More information: We have designed and tested the protocol to work even if the robot is not perfectly calibrated. However, if the calibration is off by too much, the robot may hit the walls of the well when pipetting, which can block the opening of the pipette tip.

Issue: The robot is unable to remove the pipette tips.

Solution (untested): Adjust the pick up current. See https://support.opentrons.com/en/articles/2822002-changing-advanced-pipette-settings for instructions.

More information: When performing quality controls, a specific set of tests resulted in the pipette consistently having problems ejecting the pipette tips. After contacting the support, they recommended trying to adjust the pick up current. However, the problems disappeared and never reappeared before we could test solution so it is thus untested.

Issue: When running a protocol manually over SSH from a python subprocess (not using the "OT-2 Protocol Selector" program), the calibration is off even after re-calibrating the robot.

Additional information for expanding the scope of the program, not meant for normal use.

Solution: The connection must be made in an interactive, login shell in order for the robot to initialize correctly and thus be able to find the calibration files.

Simply using `ssh -i <key> <login> <command>` will not work, instead use `ssh -i <key> <login> -t "sh -lic" <command>`

Problems starting protocol runs

The following issues a solutions refer to when the robot passes the SSH check ("Connection OK") but fails to run after pressing the "Run protocol" button.

Issue: The robot says it is paused and to close the door to continue, despite the door being closed.

Solution: Make sure that the top cover is also properly closed. Both the front door and the top cover must be properly closed (press against their respective buttons) for the robot to recognize the door as closed.

More information: The pause function constantly polls the built in door function that reports if the door is closed or not. However, the function does not distinguish between the front door and the top cover and simply reports the door as open if any of the are open.

Issues: The robot performs some initial movements after starting a run but then stops and outputs and error.

Solution: This most likely is caused by either the wrong pipettes being attached, or the magnetic module not being turned plugged in and turned on.

Setting up on a new computer

To use the "OT-2 Protocol Selector" program on a new computer, some step must first be taken.

1. Download python. It is recommended to use version 3.9 since we had issues running the opentrons package with version 3.10 of python.
2. Install the opentrons package with "pip install opentrons" on the command line.
3. Download Opentrons App from https://opentrons.com/ot-app/
4. Create a new SSH key. Instructions for this can be found on the Opentrons website: https://support.opentrons.com/en/articles/3287453-connecting-to-your-ot-2-with-ssh
   4.1. After creating a key, move it to C:\Users\USERNAME\Opentrons.
5. Download all files from the Github.
6. The program can either by launched by running main.py file directly from the installed directory, or a desktop shortcut can be made.

Basic principles of the program

In this final section of the manual, we will briefly describe the main structure of the program in case any modifications are to be made in the future.

User interface

The program is divided into several python files, with the main.py file being responsible for the user interface. The user interface in turn is divided into several classes, each responsible for a separate window. At first a Selector() object is created, which then creates either a qPCR_protocol_config() or Bead_protocol_config() object depending on which protocol type is selected. These objects are used to construct a new protocol by allowing the user to either select a finished layout in .csv format in the case of qPCR or to manually enter protocol parameters in the case of SPRI bead cleaning. The protocols are constructed by calling replace_values() or replace_values_qpcr(), both of which create protocols by combining the parameters provided by the user with an existing template file.

One the protocols have been made, a Checkbox() object is created in a new window. This gives instructions to the user for the necessary preparations required before the protocol can run and provides an image of how to load the deck on the robot. In the case of qPCR, it also provides instructions on the order to fill each tube rack with mastermixes, standards and samples. Once the preparations are complete and the program has passed a preliminary SSH check, the protocol can be uploaded and ran on the robot.

To allow pausing the protocol by opening the door, the GPIOs must be freed beforehand by stopping the opentrons-robot-server. This is done when pressing the “Check Connection” button on the Checkbox window if the connection does not fail. However, when the opentrons-robot-server is closed, the official Opentrons app will have problems connecting to the robot and it is thus important to also restart the opentrons-robot-server when finished. This is done by the “Exit” button on the same window as the protocols are started from.

Protocols

Both the SPRI bead cleaning and the qPCR protocol follow the same basic principles of passing user input from the UI to either the replace_values() or the replace_values_qpcr() function, which appends the provided parameters to a copy of a template (the blueprint files). The resulting protocol is saved as dna_cleaning_output.py and qpcr_output.py respectively.

While the general principles are the same, the SPRI bead cleaning and qPCR protocol differ somewhat in the details.

SPRI beads DNA purification

The template for SPRI bead cleaning consists of two separate blueprints. The main blueprint is used when the user inputs between 9 and 96 samples. The other blueprint is a specialized version of the main blueprint for when only a single column of samples are used, with the goal of reducing the amount of pipette tips used.

Both blueprints only use labware and instruments predefined by Opentrons. On the other hand, Opentrons’ predefined robot commands have some limitations and to overcome them to make the protocol as precise as possible, we wrote two custom functions. The first definition is a mix function which aspirates liquid at the bottom of the well and dispenses liquid at the top of the well to mix the sample as

good as possible. The second definition is a stepwise dispense which dispenses the ethanol in several steps. It divides the amount of ethanol into fractions and dispenses each fraction with increasing height from the bottom with the goal of providing a gentler dispensing without contaminating the pipette tip.

qPCR preparation

The protocol construction is more complex in the case of qPCR, since the provided in put is in the form of a .csv file and not plain numbers. Each unique mastermix, sample and standard get provided a well on a tube rack as a source well, which is saved in the form of dictionaries. Another set of dictionaries saves the destination wells on the PCR plate where each mastermix, sample or standard is to be dispensed.

The qPCR protocol uses custom labware for the tube racks, but only predefined robot commands are used.

Additionally, there also exists the ability the create a .txt file that summarizes which mastermixes, samples and standards are needed and in which well on which tube rack to place them. This file is placed in the same folder as the .csv the used provided and the protocol is based on. The file can then manually be sent to a printer for printing, with the purpose of making it easier to prepare all the necessary mixtures needed for the run.