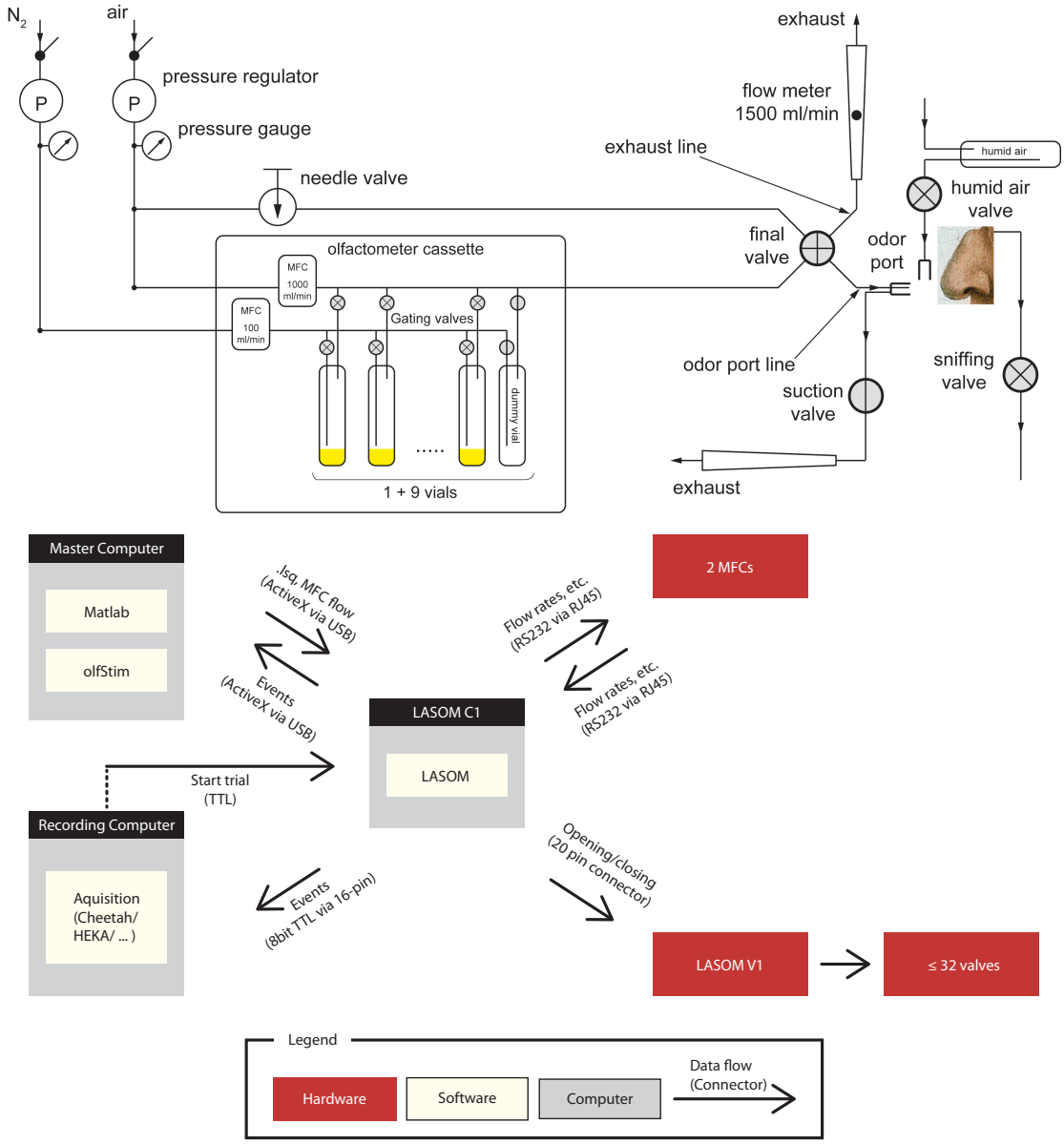
olfStim Documentation

Table of Contents



List of hardware

Setting up the hardware

Connecting the valves

LASOM makes assumptions about which valves are connected to which connectors on the LASOM2_V1 board (the board with the connectors for the valves). Therefore always connect the valves, which gate the dummy vial (they're normally open), to V1 and V2. From V3 to V20 connect the other odor vial gating valves. Connect the gating valves from the N2 stream to the even-numbered connectors, and the gating valves from the air stream to the odd-numbered

connectors. The two valves gating one vial have to be connected to two consecutive connectors.
Example: the valve gating the air stream for the first vial has to be connected to V3, and the valve gating the N2 stream for the first vial, has to be connected to V4. For the second vial, the gating valve for the air stream goes into V5, and the N2 equivalent into V6.

The other valves (final valve, suction valve, sniffing valve, or custom valves you want to control) go into valve connectors V21 to V32. You have to define the mapping of valves handled by the olfStim software to the connectors on the LASOM2_V1 board. For the default valves in olfStim the mapping is as follows:

Valve	**Connector**
Final valve	V31
Sniffing valve	V32
Suction valve	V28
Humidity valve	V27

These definitions are saved in the olfStim/lsq/core.lsq file, section "Configuration Parameters".

The definition looks something like this:

Param, finalValveIndex, 31;

If you want to change the mapping for the default valves just change the numbers in the parameter definition. 31 in this example means that finalValve is connected to connector 31.
If you want to add new valves and control them through olfStim, read on in the section for developers.

Connecting the Mass Flow Controllers

There are two mass flow controllers, one for regulating the flow of the air stream, the other one for the N2 stream. olfStim currently only supports mass flow controllers (MFC), which can be directly controlled by LASOM control board. These currently include Bronkhorst-type mass flow controllers (MFC), using the digital RS232 interfaces. Tested were the Bronkhorst MFCs IQ-Flow. See parts list for details.

There are two RS232 connectors on the LASOM_C1 board: J21 and J22. The MFC # 1 goes into J21 the MFC#2 goes into J22. OlfStim expects MFC#1 to be the air regulating MFC, and MFC#2 to be the N2 regulating MFC.

MFC #	**MFC type**	**Connector**
1	Air	J21
2	N2	J22

Olfactometer - computer connection

Get the latest LASOM package, which includes the driver

Open a command line, go to the RP Metrix foler from the LASOM package and execute the following statement:

regsvr32 lasomx.ocx

You should receive a success pop-up message.

Required testing

Before starting to use the olfactometer with olfStim some initial testing has to be done to make sure, everything works as expected. For this use the LASOM control software "LoadBoard" which can be downloaded from the RPMetrix ftp server.

Go to:

Understanding the sequencer code

Odor,slaveIndex,odorValveIndex	Command will open the two valves gating the vial defined by odorValveIndex and will automatically close the two valves gating the dummy vial in the slave defined by slaveIndex. ATTENTION: As the two valves gating the vial #2 are connected to V3 and V4, the odorValveIndex of the first vial is 2. The reason is because the dummy valves are connected to connectors V1 and V2. Therefore all valve indices are shifted by 1.
Gate,slaveIndex,valveIndex	Command will open the valve connected to the connector defined by valveIndex on the slave defined by slaveIndex. Gate,1,3 will open the valve connected to connect V3 on slave 1.
StartTimer, 2	Starts an internal Timer at timer index 2 (timer indices 2-7 are generally usable, timer indices 0 and 1 are reserved for loops.
WhileAlways SinceTimer, 2, 3	Enter an infinite while loop, for every iteration, compute the relative time since the timer was started for timer index 2, write that relative time into timer index 3. Check whether the time in index

CheckLapse, 3, someTime IfLapse, 3, @lapsedOut EndWhile	3 exceeds the time in ms defined in someTime, store the result. IfLapse checks whether timer index 3 did exceed the defined time value, if so the sequencer jumps out of the while loop to label @lapsedOut.

For users

Installation

Download olfStim from the github repository:
https://github.com/lorenzpammer/olfStim

Add the olfStim folder to the Matlab path. Only select the parent folder, as all necessary subfolders will be added to the path once olfStim is being called.

Necessary customization

Even though it was a design goal to make olfStim work out of the box. Some customizations cannot be avoided.

Odorant library

All odorants you want to present have to be entered into the odorLibrary, which is just a text file with definitions for each odorant. The purpose is, that all information, which characterizes an odorant is entered once for each odor in use. The file is located under olfStim/odorLibrary/odorLibraryGenerator.

Let's say we want to add Cineole to our library of odorants. We'll first add the odor name to the list of odor names:

```
odorNames = {'Benzyl alcohol','Butanone','Camphor','Citral', ...
    'Eugenol','Geranyl acetate','Hexanal','1-Hexanol','2-
Hexanone','Isoamyl acetate', 'Vanillin','Turtle food','Paraffin
```

```
oil','Mineral oil','Ethanol', 'Water','Empty', 'Cineole'}; % Cell
array of strings with names of odorants in Library
```

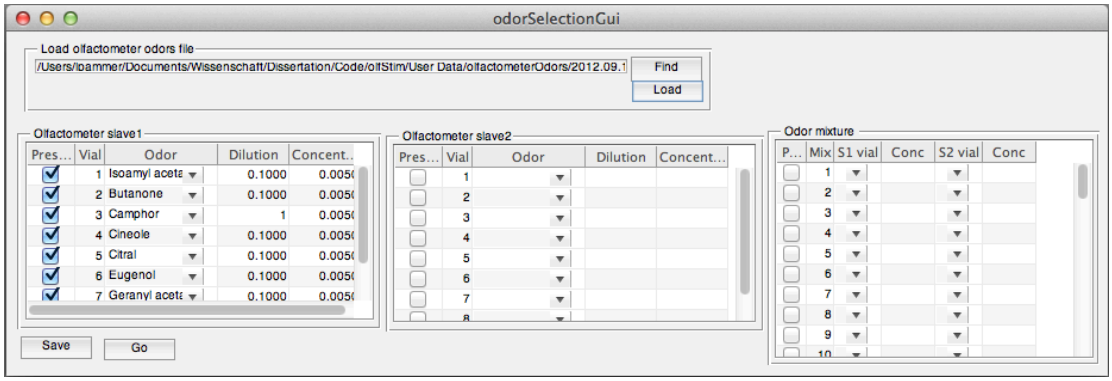
Then we'll add an entry to the list of odors. To do this just copy the template, which you can find at the top of paragraph C in the odorLibraryGenerator.m file, paste it to the end of the file and update the input to each field for the odorant you are using.

```
odor = 'Cineole';
for i = 1 : length(odorLibrary)
    if strcmp(odorLibrary(i).odorName, odor) % at the respective odor
write the following properties to the structure:
        odorLibrary(i).iupacName = '1,3,3-trimethyl-2-
oxabicyclo[2,2,2]octane'; % the name of the molecule following IUPAC
convention as a string
        odorLibrary(i).CASNumber = '470-82-6'; % the CAS registry
number identifying the molecule as a string
        odorLibrary(i).producingCompany = 'Sigma'; % the name of the
company that produced the odorant: 'Sigma', 'Roth', etc.
        odorLibrary(i).odorantPurity = 0.99; % purity of the molecule
given as a fraction, usually above 0.95: 0-1
        odorLibrary(i).state = 'liquid'; % string describing whether
the state of the molecule is liquid or solid: 'liquid' or 'solid'
        odorLibrary(i).odorantDilution = 0.1; % volume fraction
(v(odorant)/v(dilutive solution)) of odorant in the vial after
diluting it with water or oil: 0-1
        odorLibrary(i).dilutedIn = 'Paraffin oil'; % in which
solution the odor was diluted: 'Water' 'Paraffin oil' 'Mineral oil'
or [] if no dilution
        odorLibrary(i).concentrationAtPresentation = 0.005; %
concentration as volume fraction (v/v) of saturated headspace, which
is presented to the animal: 0-1
        odorLibrary(i).inflectionPointResponseCurve = []; %
concentration (v/v, of saturated head space) of presented odorant at
which the response curve as measured in olfactory nerve has its
inflection point.
        odorLibrary(i).vaporPressure = [253.3 25]; % vapor pressure
in Pascal @ 25?C. For mixtures use Raoult's law. To convert mmHg into
Pascal multiply by 133.322
    end
end
```

Running olfStim from the gui

In the Matlab command line type
`initOlfStim`

This will result in opening the odorSelectionGui to open.



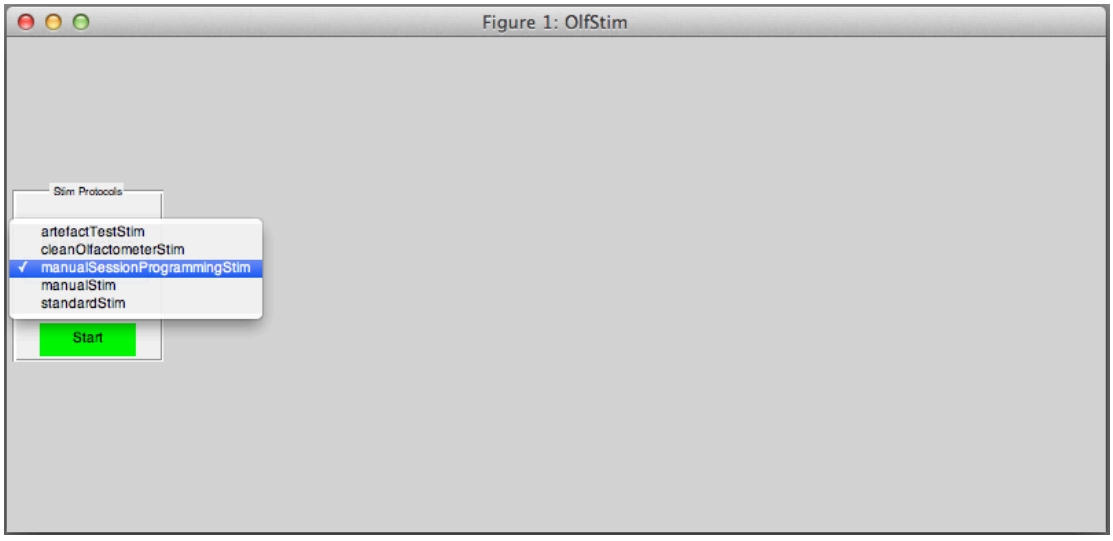
For every slave, choose which odorant is loaded into each connected vial. In the first column check whether you want to use this odor in the session. In the third column choose the odorant contained in the vial from the dropdown menu – the vial is specified by the vial number in the second column.
Make sure that the dilution of the odorant in the vial (eg you diluted Cineole in Paraffin oil) is specified correctly in the 4th column. For the standardStim protocol it is necessary to set the concentration at presentation (v/v) in the 5th column.

In order to avoid having to enter the loaded odorants every time, you can save the settings to a file by clicking the "Save" button. Previously saved settings can be loaded by specifying the location of the file with the "Find" button and loading it with the "Load" button.

NOTE: odor mixtures can not yet be used.

Once the Gui contains the information which odor is contained in which vial click the "Go" button.

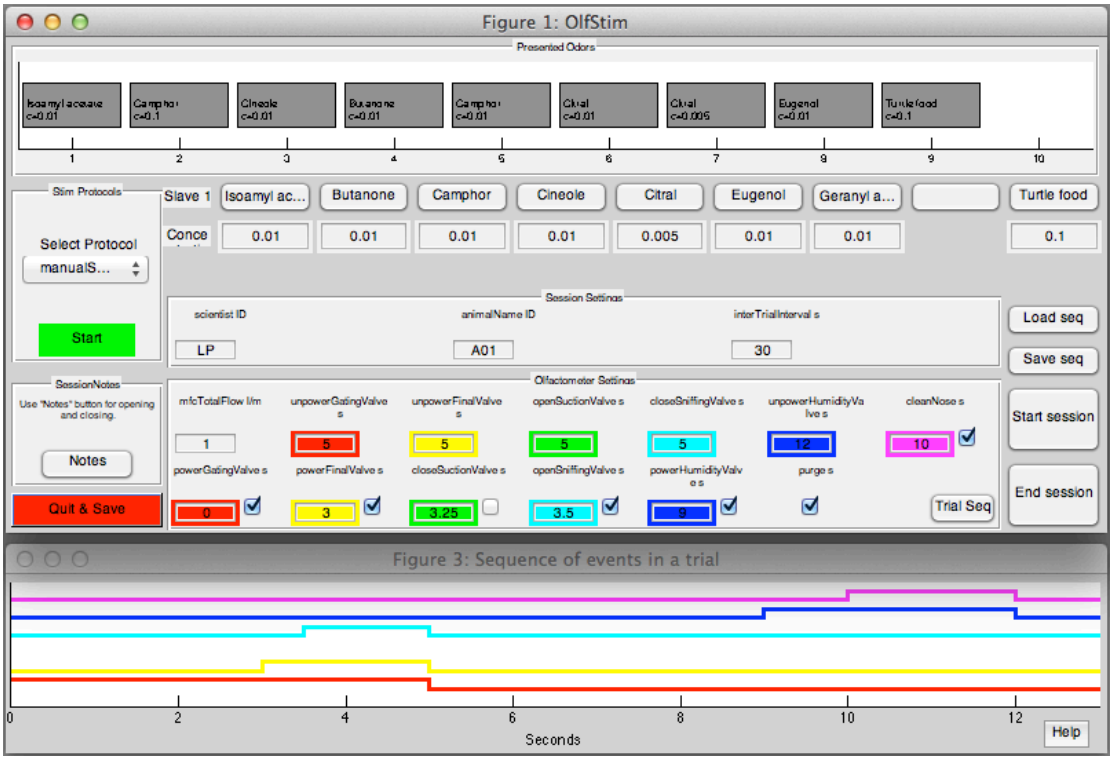
The main gui will now open and you have to choose which stimulation protocol you want to use. See details about the protocols below. Then click the "Start" button.



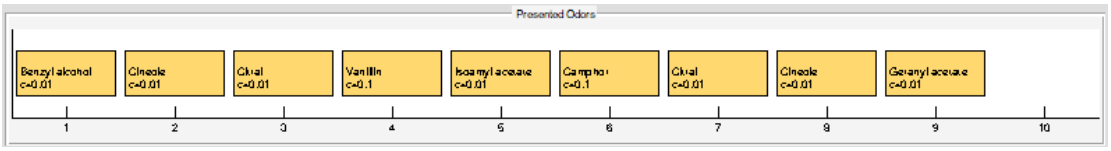
This results in the gui being populated by all features of the chosen protocol. You are now ready to start presenting odors! For the explanation of the protocols' features read the protocols section in this document.

Features

In this section olfStim's general features are documented.



Progress panel

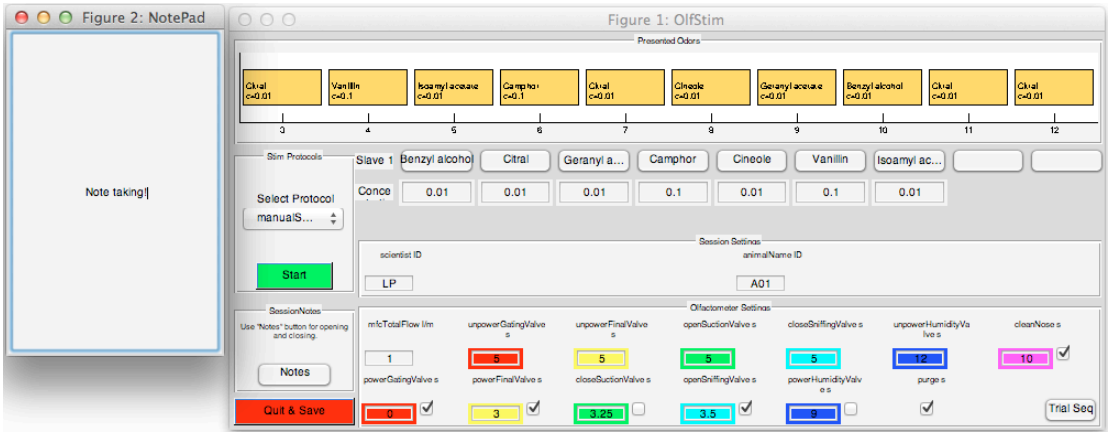


The progress panel is at the top of the GUI and shows the history of presented odors, identifying each trial with the odor name and the concentration.

Notes

Clicking on the "Notes" button on the left lower part of the GUI will open a new figure – the note pad - for note taking. In this figure you can write your notes for the current trial. When the trial is triggered the notes are extracted and

associated with the trial in the output data structure. At the same time the figure will be cleaned of the old notes.
Use the “Notes” button both to open and close the note pad.



Olfactometer Settings

In the olfactometerSettings panel the times of valves opening & closing can be entered manually for each trial. Also multiple opening and closing times are allowed. This can be done by entering two values in square brackets (matlab vector notation). For instance entering [3 6] in the open sniffing valve and [5 8] in the close sniffing valve field will result in the sniffing valve opening at 3 s, closing at 5s, opening again at 6 s and closing again at 8 s.
Note however that at the end of the trial the resting state has to be reestablished. Which means you have to close/unpower every valve you open/power.

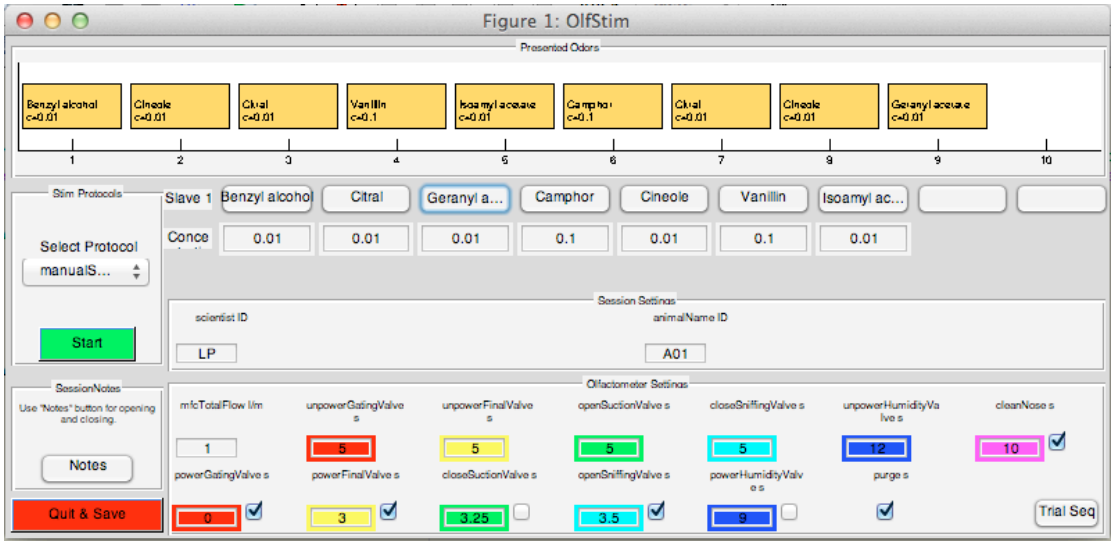
Protocols

Cleaning

Start by filling all vials with Ethanol.

manualStim

The manualStim stimulation protocol is simple and a good starting point to test the olfactometer but also for exploratory experiments. It allows you to set the concentration of the odor presentation, control the flow rate and the opening and closing times of relevant vials and then trigger trials by clicking.



manualProgrammingStim

Running the program in test mode

In the Matlab commandline type
`initOlfStim('testing')`
or
`initOlfStim('test')`

This will start the test version of olfStim, which does not require the computer to be connected to an olfactometer. Most features not directly linked to olfStim-olfactometer interaction are active and can be tested. The test mode is useful if you want to start playing with the software but don't have an olfactometer connected, or if you're coding/hacking olfStim and want to test your code on your development machine.

Scripting olfStim

For Developers

Set air and Nitrogen pressure to 2 bar.