Appendix: Instruction Manual for the Smellevision

An Olfactory Virtual Reality System

Warner Bros., 1944

By Brad A. Radvansky

Laboratory of Daniel A. Dombeck

Northwestern University

August, 2020

“It will never work!”

-Carl Stalling

Table of Contents

Introduction……..196
FAQs……198
Building the System…….200
Hardware……..200
Olfactometer……200
List of Parts……..200
Operating Principles…….201
Building Instructions……206
Plumbing……..206
Wiring……..208
Virtual Reality Controller…….209
List of Parts……..209
Operating Principles…….209
Building Instructions……211
Other Task-Related Hardware……..212
Software……...212
VR Computer……...212
Operating Principles…….212
Setup……212
Logging Computer……...212
Operating Principles…….213
Setup……213
Validating the System……..214
Commanding the MFCs……...214
Checking for Leaks……..216
Choosing Odorants……...217
Single Chemical……...217
Chemical Stability……218
Volatility……..218

Preparing an Odorant Saturation Chamber……...219
Choosing Concentrations…….220
Testing the Duration of Odorant Delivery……222
Testing the Speed of Odorant Delivery…….222
Implementing the System…….227
Designing an Olfactory Virtual Landscape……...227
Designing a Task……..228
Generating an Olfactory Virtual Landscape…….229
Reading from the Controller…….230
Logging the Experiment Variables……...232
Moving Through the Olfactory Virtual Landscape……...233
Generalizing to Two Dimensions…….235
Adding Non-Olfactory Cues……236
Validating the System for Behavior……..238
Fine-Tuning the System……...239
Position-Predictive Algorithm…….241
Position-Amplitude-Predictive Algorithm……...242
Upkeep…….248
Checks to Perform Each Time the System is Used……248
Checks to Perform Periodically……249

Introduction

Welcome to Smellevision, where the world is made of smells. This is the world's first system to control odorant concentrations as continuous variables across space in virtual reality. This is done by using a feedback loop to rapidly deliver odorants to the nose while the subject controls movement through virtual space. This may sound straightforward in principle, but in practice it is quite difficult to reliably control volatile chemicals at this level of precision. This system is guaranteed to work for a particular set of parameters (mouse navigating on a treadmill smelling methyl valerate and α-pinene odors). This system is likely be applicable across a wide range of species, behaviors, controllers, and odors, but the burden to is on the user to establish this. Building and validating this system is not a casual undertaking, especially for projects beyond the established parameters. It will require several expensive pieces of equipment, a substantial time investment from one or more skilled workers, and a rigorous commitment to performing the proper design and validation procedures. It is imperative that the user follow this manual exactly. One may be tempted to cut corners in various ways, such as not measuring the odorant concentrations, not monitoring the airflow, not checking the system for leaks, not refreshing the odorants from session-to-session, not personally smelling the delivered odors before starting each session, or interpreting the subject's behavior as evidence that the system surely must be working and no longer requires monitoring. Such shortcuts can allow technical issues to go unnoticed, which will in turn confound the results of behavioral experiments using this system, ultimately wasting people's time and causing serious detriment to science. The Smellevision was invented for good – if it were to fall into the wrong hands, there is no telling what havoc it could wreak upon the world. Please use it wisely.

If you accept this responsibility and do successfully build this apparatus, please make sure to cite at least one of the following official references:

- Radvansky, B.A. & Dombeck, D.A. (2018). An olfactory virtual reality system for mice. Nature Communications 9, 839.
- Dombeck, D.A. & Radvansky, B.A. (2019). Introduction of olfactory cues into a virtual reality system. United States Patent Application Publication US 2019/0346865 A1.

The user may come from a background in olfactory science, virtual reality systems, both, or neither. Thus, this manual attempts to cover the relevant basics of both fields, and is intended to be accessible to anyone with basic proficiency in computer programming. Hopefully this is indeed the case. For technical advice, contact Brad Radvansky (radvansky@fastmail.com). I wish you success in using this system to produce meaningful contributions to society.

Thanks to Tom Bozza, Adam Dewan, Annika Cichy, and Alicat Scientific for technical assistance and helpful discussions. This project was financed by The Klingenstein Foundation, The Whitehall Foundation, Northwestern University, The Chicago Biomedical Consortium with support from the Searle Funds at The Chicago Community Trust, The NIH (1R01MH101297) and NSF (CRCNS 1516235).

FAQs

How can I know that this system can work?

This system has been published as a methods paper in a peer-reviewed scientific journal (Radvansky and Dombeck, 2018), and a formal patent application has been filed (Dombeck and Radvansky, 2019). These documents establish that:

- The Smellevision can precisely control odorant distributions during behavior in virtual reality.
- Mice in the Smellevision can use odors to guide their behavior in virtual reality.
- This behavior engages neural mechanisms of spatial cognition.

Who cares about mice? Does this work for humans?

Many people care about mice! And mice care about each other. There is evidence that humans can navigate using odor (Porter et al., 2006), and that odors can engage human brain mechanisms of spatial cognition (Bao et al., 2019), so it is likely that olfactory virtual reality could be immersive for humans. But this has yet to be validated.

Do I have to lease the patent to build a smellevision?

Currently, the patent is pending, and this system is not protected by any intellectual property rights.

Why are you trying to patent this technology while also giving away these instructions for free?

Our goal is to make this technology accessible and useful for people. Perhaps the best way to do this is by encouraging a tech company to mass-produce and package it. Or perhaps the best approach is to provide detailed instructions for individuals to build it on their own.

Why is this manual attached to a PhD dissertation?

The dissertation describes every experiment that has ever been done using this system, why these experiments are important, and how the technology of olfactory virtual reality has opened new areas of neuroscience research. The dissertation and manual complement one another.

Can the Smellevision control every kind of odor?

No. The chemistry of odorants is complicated, and not every odor is suitable for this technology. See the Choosing Odorants section.

Is it expensive to build a smellevision?

Yes, you should get someone else to pay for it.

Can the smellevision be synchronized with television/audio/etc.?

Yes, this is relatively straightforward. See the Adding Non-Olfactory Cues section.

Isn't there already an invention from the 1930s called Smellevision?

That one was called Smell-O-Vision, this is totally different.

Have people actually frequently asked these questions?

No, they are all made up except one.

Building the System

Hardware

This manual describes a setup to control two odors in virtual reality. With minor modifications (and more mass flow controllers), this could easily be scaled up for many odors. Feel free to update any parts that have grown outdated over time.

Olfactometer

List of Parts

1 optical table (Thorlabs, optional)

1 reliable source of pressurized air

1 long coil of 1/4 inch nylon tubing (several meters)

1 set of valves and junctions for the nylon tubing (NResearch)

1 pressure regulator

1 gas purifier (AFS Purifier II, Chromatography Research Supplies)

2 mass flow controllers (MFCs, Alicat MC-100SCCM-D, flow rate 0–0.1 L min^{-1})

1 mass flow controller (MFC, Alicat MC-15SLPM-D/10 M, flow rate 0–1 L min^{-1})

1 box of threaded Teflon fittings (NResearch)

3 MFC outlet screws to fit the threaded Teflon fittings

1 roll of Teflon tape

1 coil of 1/32 inch Teflon tubing (couple of meters)

1 coil of 0.002 inch Teflon tubing (couple of meters)

1 case 40-mL amber glass vials with rubber membrane caps (Fisher)

1 bottle of mineral oil (~1 L)

1 bottle of odorant 1, such as methyl valerate (Sigma-Aldrich)

1 bottle of odorant 2, such as α-pinene (Sigma-Aldrich)

1 jar of 3 mm diameter soda-lime glass beads (Sigma-Aldrich)

1 Teflon mixing block (custom-made, see below)

1 Teflon nose chamber (custom-made, see below)

1 Photo-ionization detector (PID, Aurora)

1 tank of compressed nitrogen

1 box of kimwipes

1 wide-mouthed container for water

1 data acquisition card (National Instruments PCI-6229, more for more odors)

1 computer with Matlab (or your preferred VR language), called the "VR computer"

1 Digidata1440A (Molecular Devices) data acquisition system

1 computer with Clampex and the Digidata installed, called the "logging computer"

Note that the MFCs and PID are expensive and somewhat fragile. These must be handled frequently, so handle them with care!

Operating Principles

The purpose of this olfactometer is to quickly and reliably deliver odorants of particular concentrations to the subject's nose, and to quickly change these concentrations when desired. This olfactometer differs from previous designs in several ways. First, it is designed to deliver odorants continuously rather than as discrete pulses (Shusterman et al., 2011). Second, similar to one existing olfactometer (Kim et al., 2015), it is designed to control odorant concentrations as continuous variables, rather than as binary on/off stimuli (Lovett-Barron et al., 2014; Poort et al., 2015).

The olfactometer is driven by a constant source of pressurized air (Figure 1a). This air is filtered to remove contaminants such as moisture (Figure 1b). The filtered air is split to three tubes, each leading to a mass flow controller (MFC, Figure 1c-e). The MFCs are the workhorses of the olfactometer. Their purpose is to receive voltage signals from a computer, then produce an output airflow rate that is linearly related to that voltage. This voltage will later be synchronized with virtual reality to control odorant concentrations as spatial variables (see Running the System). Two of the output tubes of the MFCs lead to odorant saturation chambers, which are essentially bottles filled with glass beads and liquid odorant solutions (Figure 1f,g). The third MFC output tube is called the carrier stream, and leads to a mixing junction (Figure 1h), where it will meet the two odorized air streams.

The design of the odorant saturation chambers is not trivial, and may need to be modified for new odorants (see Validating the System). Since the olfactometer must operate constantly, the head space (gaseous top section) of these chambers must remain saturated for many minutes to hours. This involves a complicated relationship between many factors including odorant concentration, vapor pressure, liquid volume, head space volume, oxidation, and air-to-liquid surface area. There several procedures to optimize these factors (see Validating the System). The output tube of the odorant saturation chambers leads to a mixing junction to meet the carrier stream.

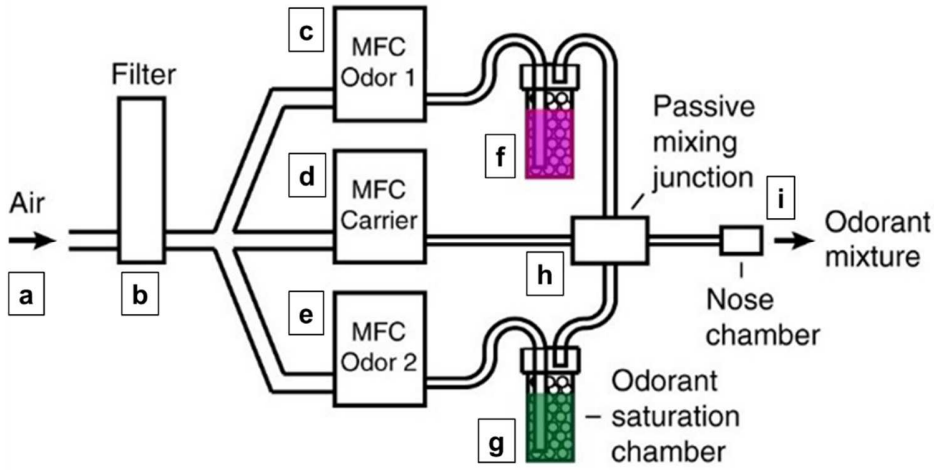

Figure 1. The olfactometer.

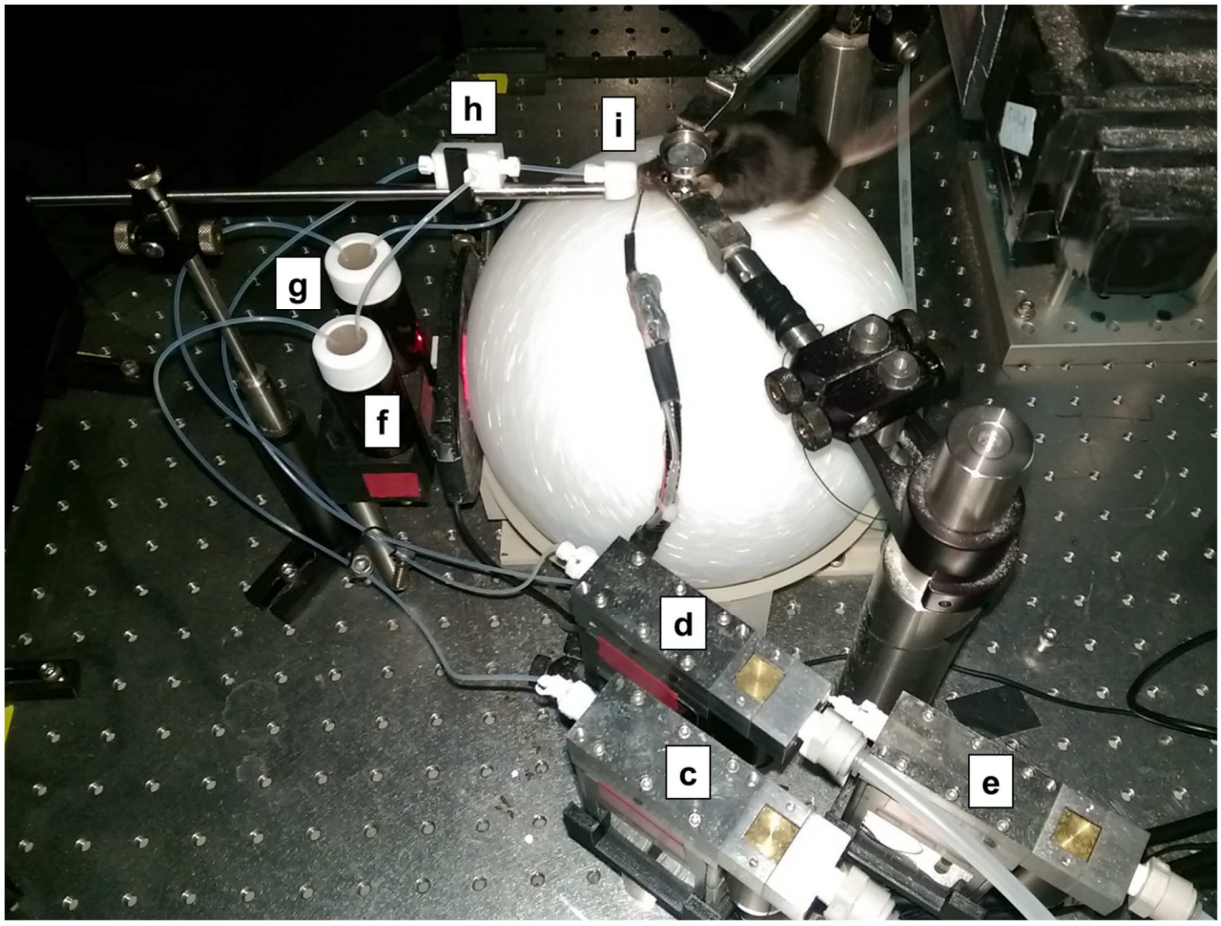

Figure 2. Photograph of the olfactometer, controller, and subject.

This mixing junction is simply a passive piece of Teflon with holes drilled in it. There do exist turbulent mixing chambers for purchase which the user could consider for mixing the odorants very thoroughly, but such a device is likely to sacrifice speed of odorant delivery, and thus olfacto-spatial resolution in VR. The carrier stream can serve several purposes. 1.) It can be updated dynamically to maintain a constant final flow rate at the nose, thus controlling for variation in wind as a confounding stimulus. 2.) On the contrary, it could be used to simulate odorless wind. 3.) It can dilute the odorants by orders of magnitude, which can smell quite strong at full concentration, which could lead to problems of saturating the subject's olfactory receptors or producing noxious aromas.

The mixing junction output tube leads to a nose chamber (Figure 1i), which is simply another piece of Teflon with holes drilled in it. It is crucial that this nose chamber be designed such that both of the subject's nostrils are within the cavity of the nose chamber (but obviously it is not recommended to make an airtight connection between the nostrils and the olfactometer). The purpose of placing the nostrils within the nose chamber is to create a micro-environment in which all of the air entering the nose comes from the olfactometer and thus the odorant concentrations detected by the subject are roughly equivalent to the odorant concentrations delivered by the olfactometer, with no contamination from the ambient air. This setup offers the added advantage that no odorant removal system such as a vacuum is needed. Since the subject's nostrils are within the nose chamber and the olfactometer flow is constant, the subject only smells exactly what comes from the olfactometer at all times. Nonetheless, it is recommended to install ventilation and filtration systems in the room to minimize ambient odorant levels.

In virtual reality (VR), the subject should be able to move reasonably quickly and freely, and should be able to remain immersed for extended time periods. Thus, olfactometer needs to

operate rapidly and consistently over long durations. In practice, to operate rapidly means to minimize the time between commanding the MFCs (Figure 1c-e) and receiving the odorants at the nose chamber (Fig. 1i). To operate over long durations means to maintain a reasonably constant relationship between the command of the mass flow controllers and their physical concentration at the nose chamber over many minutes to hours. These parameters of speed and duration depend both on the physical construction of the olfactometer and the chemical properties of the odorant. This section describes how to build a fast olfactometer with reasonably good speed and duration for two particular odorants. For new odorants, the user must follow the procedure in the Validating the System to ensure that these parameters are acceptable.

Obviously, a fluid will travel from end-to-end through a short tube faster than a long one. Thus, the lengths of all tubing after the MFCs should be minimized, while still leaving reasonable slack for the user to replace the odorant mixing chambers, troubleshoot the MFCs, and perform water-immersion leak checks of all junctions (see Validating the System). A fluid will also travel faster through a narrow tube than a wide tube. Thus, the user's intuition may be to use the narrowest possible tubing for the whole olfactometer. This is a bad idea, and can cause two serious problems: 1.) Back-pressure and capillary action, which can push the odorant solution backward into the olfactometers thereby damaging them. 2.) Strain on the MFCs to regulate flow properly at the high pressure caused by the narrow tubing. It is good practice to always monitor the MFC flow rates to identify such malfunctions, either by observing their digital displays, or better by tracing their flow rate analog waveforms using a digidata system. See the MFC instruction manuals for information about cleaning and pressure limitations, and output wiring. It is recommended to use exactly the tube widths suggested in this manual, and also making sure that the holes in the mixing junction and nose chamber are at least the width of the corresponding tubing. Because the carrier stream

operates at an order of magnitude higher flow rate than the odorant streams, it is recommended to use relatively wide tubing from the carrier MFC to the mixing junction and from the mixing junction to the nose chamber, compared to the tubing width from the odorant saturation chambers to the mixing junction. This is to create a "path of least resistance" in which the carrier stream flows toward the nose chamber rather than into the odorant streams. Similarly, the odorant streams must never reach zero flow, or else the carrier stream could cause odorant solution backflow. It is recommended to define minimum odorant flow rate as 1 mL/min.

One computer controls the virtual reality session and the olfactometer using a programming language such as Matlab and a data acquisition card. A second computer synchronizes and logs the data from each timepoint of each session using Clampex software and Digidata hardware. This second computer is not strictly necessary, and in theory could be replaced by other data synchronization and storage methods. However, Clampex and Digidata are preferred because in real-time they can visualize the waveforms of each MFC flow rate, the PID signal, behavioral parameters such as virtual position, velocity, and bearing, task parameters such as rewards, and recording methods such as microscope imaging frame times. If this hardware is not available, then at least the waveform of each MFC should constantly be monitored using an oscilloscope. When choosing a data acquisition card and Digidata, make sure that they have enough analog input and output channels to run the experiment, as well as any necessary digital or counter channels.

Building Instructions

Plumbing

1. Build the passive mixing junction. Drill 1/32 inch holes through the axes of a ~(½ by ½ by ½) inch Teflon block to make a 4-way intersection of tunnels. Machine the surfaces of the holes to tightly fit threaded Teflon fittings.

2. Build the nose chamber. The size of this part depends on the species of the subject. Drill a 1/32 inch hole through a Teflon block. Machine one end to tightly fit a threaded Teflon fitting. Machine the other end to be just large enough that the subject's nostrils are completely beyond the plane and thus completely within the nose chamber.

3. Clear an area on the optical table, or wherever you choose to build this system. The optical table has the advantages of stability and screw holes to mount the parts, but will require additional hardware such as posts and screws. It is recommended to mount the parts to the table once they are installed. This manual assumes the installation to be permanent. Consider the space that the subject, controller, and other parts such as a visual display will occupy, and choose carefully where to build the apparatus. All parts of the olfactometer should be easily within reach of the user. The MFCs should be mounted higher than the odorant saturation chambers or even upside-down to prevent backflow of the odorant solutions into the MFCs.

4. Run nylon tubing from the pressurized air source to a valve.

5. Run nylon tubing from the valve to the pressure gague.

6. Run nylon tubing from the pressure gauge to a second valve.

7. Open the first valve and close the second. Set the pressure gauge to 20 PSI.

8. Run nylon tubing from the second valve to the gas purifier.

9. Run nylon tubing from the to two Y-splitter junctions to split it into 3 tubes.

10. Attach the 3 tubes to the inlets of the 3 MFCs.

11. Attach each MFC outlet to a MFC outlet screw, then to a threaded Teflon fitting, both secured with Teflon tape.

12. Prepare two vials with membrane caps by poking 2 needle holes spaced apart in each lid.

13. Run 1/32 inch Teflon tubing from MFC Odor 1 (max flow rate 100 mL/min) through the lid of Odor Vial 1, to the bottom of the vial

14. Run 1/32 inch Teflon tubing from MFC Odor 2 (max flow rate 100 mL/min) through the lid of Odor Vial 2, to the bottom of the vial

15. Run 0.002 inch tubing from poking through the very top of Odor Vial 1 to the mixing junction.

16. Run 0.002 inch tubing from poking through the very top of Odor Vial 2 to the mixing junction.

17. Run 1/32 inch tubing from the MFC Carrier outlet to the mixing junction. Secure all inlets to the mixing junction with threaded Teflon fittings and Teflon tape

18. Run 1/32 inch tubing from the mixing junction to the nose chamber. Secure both junctions with threaded Teflon fittings and Teflon tape. The mixing junction and nose chamber can be mounted to a horizontal steel rod, which can be mounted to the table by a vertical steel rod. It should be possible to firmly secure the nose chamber around the nose without touching it. It is helpful to mount the mixing junction and nose chamber firmly so that the junctions do not come loose from strain.

Wiring

1. Plug in the MFCs. It can be convenient for the three MFCs to share one designated power strip that can be switched off when not in use.

2. To optimize each MFC's control speed, press its buttons and navigate to the parameters window. Set each MFC's parameters to:

   MFC Odor 1: P = 00050, I = 00500, D = 00080
   MFC Odor 2: P = 00050, I = 00500, D = 00080
   MFC Carrier: P = 00100, I = 02000, D = 00080

   These parameters were chosen by technicians of Alicat Scientific for this custom application. For parameter definitions, see the instruction manual of the MFCs.

3. Install a computer with Matlab software (or your preferred software with data acquisition capability) and a data acquisition card.

4. Install a second computer with Clampex software and Digidata hardware.

5. Find the wiring diagram of the MFC cables in their instruction manual. Strip the cut ends of each cable to expose each wire. The wires of significance are:

   - Analog monitoring voltage from the MFC to the computer to record its flow rate
   - Analog monitoring ground
   - Analog command voltage from the computer to drive the MFC
   - Analog command ground

6. Connect each monitoring wire and its corresponding ground to an analog input channel of the Digidata. If your Digidata input port is BNC, this will require attaching the wires to BNC adapters. Now the logging computer can monitor the flow rate of each MFC via the Digidata. It is good practice to label all wires with a piece of tape, and to take careful notes. These channel numbers will be necessary for implementing the software

7. For each pair of command wires and grounds, split this pair into two pairs each containing command and ground. One of these is for an analog output from the data acquisition card, and one of these is for an analog input to the Digidata. This procedure depends on the type of inputs ports used by the data acquisition card and Digidata. If both are BNC, then a BNC adapter and splitter can be used. If one is a breakout board and the other is BNC, then the raw wires can be plugged into the breakout board, then a BNC adapter soldered to short lengths of raw wires can be plugged into the same ports of the breakout board. Now the VR computer can command the MFCs and the logging computer can monitor this command. It may seem redundant to monitor both MFC command and MFC feedback, but each of these values has important uses for testing and monitoring the system's performance.

8. Label the adapter end of each MFC cable and plug it in to the corresponding MFC. Press the buttons on each MFC to set the command mode to analog voltage. Now each MFC is ready to control flow rate by reading the voltage from the command wires from the VR computer. You can try doing this using the data acquisition software that probably came with your data acquisition card. The relationship between Volts and flow rate should be linear, with 10 Volts corresponding to maximum flow rate. Make sure to open the olfactometer air valves first.

Virtual Reality Controller

List of Parts

1 floating ball mouse behavioral apparatus (see Operating Principles)

1 reliable source of pressurized air (in addition to the olfactometer source)

1 optical computer mouse

1 digital to analog (D/A) converter

List of Parts Already Installed in the Olfactometer Section

1 Optical table (Thorlabs)

1 data acquisition card (National Instruments PCI-6229)

1 computer with Matlab (or your preferred VR language), called the "VR computer"

1 Digidata1440A (Molecular Devices) data acquisition system

1 computer with Clampex and the Digidata installed, called the "logging computer"

Operating Principles

Anyone who has played a first-person video game has used a virtual reality controller, which can be defined as a device that converts the subject's real movement into movement in virtual reality. Common examples include keyboards, mice, and joysticks, but there are vast possibilities such as treadmills, accelerometers, gyroscopes, lasers, EMG leads, and body- or eye-tracking systems. The mechanism of any controller is to detect movement and transduce it into an electronic analog or digital signal that can be read by a computer. In principle, all types of controllers are compatible with the Smellevision.

The controller simply needs to be wired to the data acquisition card of the VR computer, read by the VR engine, and transformed by some physics equations to produce the proper movement in the virtual world. This section will walk through the example of a spherical treadmill — essentially a floating ball on which a head-fixed mouse can run, but same procedure applies to any controller that is a rolling ball. This procedure has already been described many times for visual VR (Harvey et al., 2009; Dombeck et al., 2010; Aronov and Tank, 2014), but for ease of access, the same procedure will be summarized here. To get a floating ball mouse behavioral apparatus, the user can build it by following previous instructions (Dombeck et al., 2007) or purchase it from a VR company such as PhenoSys. For a 1D rotary controller such as a wheel or disc, the user can simply use the spherical treadmill example and ignore one dimension. Generalizing from this procedure, the user should be able to interface any type of controller with the Smellevision. For more information on VR controllers, an excellent practical resource is "Virtual Reality Matlab Engine. ViRMEn – The Manual" by Dmitriy Aronov (Aronov and Tank, 2014).

The spherical treadmill transduces movement into an analog signal, but some controllers such as rotary devices are digital. The only substantial difference is the wiring of the controller to

the data acquisition card. Analog devices require analog input channels that directly read voltage, and digital devices require counter channels that count the number of digital pulses per sample. The user should make sure to install a data acquisition card (or several) with enough channels of each type for all devices used. Note that controllers generally require one channel per degree of freedom (e.g wheel: 1; ball: 2 or 3). Once the voltage(s) or pulses count(s) are read into the VR engine, these numbers can be fed into physics equations and transformed into the proper virtual movement.

Building Instructions

1. Choose a space on the optical table to build the controller. The subject's nose should be able to fit in the nose chamber while using the controller. This should already have been considered when building the olfactometer.

2. Attach the floating ball mouse behavioral apparatus to the optical table. Attach the pressurized air source.

3. Mount the optical mouse firmly at the equator of the ball in line with the forward direction that the subject will face. The mouse must be parallel or perpendicular to the ball, and should be as close as possible to the ball without touching it. Test this by turning on the airflow and spinning the ball in many directions. This check should be performed often in case the mouse becomes dislodged or the ball shrinks with friction over time. The mounting can be achieved using optical table posts (Thorlabs).

4. Plug the optical mouse into the D/A converter. This should convert the mouse signal into analog voltages, one for up-down (pitch of the ball) and one for left-right (yaw of the ball). The output of the D/A converter should be one wire for yaw plus its ground wire, and one wire for pitch plus its ground wire. For finer movement control, a second optical mouse can be mounted at the equator on the side of the subject to record the roll of the ball.

5. As described in the olfactometer building instructions, split the pitch wire and its ground into two pairs of wires. Plug one pair into an analog input channel of the data acquisition card and its corresponding ground. Plug the other pair into an analog input channel and ground of the Digidata. Repeat this for the yaw wire and its ground. Now, the VR computer can read both dimensions of the optical mouse, and the logging computer can log these signals. These voltages are linearly related to the subject's movement velocities in the pitch and yaw direction.

Other Task-Related Hardware

The user may wish to include other inputs and outputs to the virtual reality system, such as reward delivery, lick sensors, sniff sensors, eye trackers, or devices for stimulating or recording other modalities of sensory information such as visual, auditory, or tactile. These devices can be incorporated using similar wiring to the olfactometer and controller. As long as the device can input or output an analog or digital signal, it can be interfaced with the Smellevision.

Software

VR Computer

Operating Principles

This computer's job is to run the VR engine. This engine can be a sophisticated software package such as ViRMEn (Aronov and Tank, 2014), or a simple for loop in any programming language. At each iteration of the loop, the engine needs to read from the input devices such as the VR controller, update the VR variables such as position and velocity, and command the output devices such as the olfactometer. This process is relatively computationally inexpensive unless visual cues are used, in which case a good quality graphics card should be considered.

The VR computer needs software for a programming language with data acquisition functions. This manual provides instructions for Matlab with the Data Acquisition Toolbox.

Setup

1. Install the VR computer with the latest version of Matlab. Make sure to install the Data Acquisition toolbox.
2. If visual cues are desired, download the ViRMEn engine from virmen.princeton.edu (Aronov and Tank, 2014).

Logging Computer

Operating Principles

This computer's job is to synchronize and log the data of the VR session and other data such electrophysiology recordings, as well as visualize these data in real-time. In principle, this computer could be eliminated entirely, and the data could be directly logged by the VR computer. However, this is not recommended. Using a logging computer can be very helpful for monitoring and troubleshooting the system, as well as observing the subject's behavior. Since the odors and MFC flow rates are invisible, the user must instead use a computer to visualize their corresponding voltage waveforms online. If a logging computer is not available, an oscilloscope can do the job. The user should never run this system without visualizing the MFC flow rates online. This manual recommends using a Digidata (Molecular Devices) data acquisition system (Clampex), but other similar data logging systems should suffice.

Setup

1. Install the logging computer with the Digidata device and the latest version of Clampex.

2. In Clampex, create a protocol. Set the protocol to read each input from the VR experiment (e.g. odor 1 flow rate, pitch velocity, etc.) as well as from any external devices such as reward solenoids, recording electrodes, or microscopes. Make sure to include all MFC feedbacks as well as both odor MFC commands. While this seems redundant, it will be necessary for testing (see Validating the System).

3. Select an additional channel for the PID input. This is the device that measures odorant concentrations. It will be needed for testing (see Validating the System).

4. Select additional channels for the variables in VR that you will want to record (e.g. x position, y position, view angle, environment number, correct vs. incorrect, etc.)

5. Label each input and save the protocol.

Validating the System

All hardware and software for the Smellevision are now installed. The next step is to make sure that the olfactometer can achieve its two key purposes: to deliver odorants rapidly and reliably for long durations. The user may be tempted to skip these steps and simply sniff at the olfactometer and assume that it probably works well enough. This is not acceptable. To build a working Smellevision, these validation steps must be followed exactly.

Commanding the MFCs

Now that the wiring and software installation is complete, the olfactometer can be controlled by the programming language. This can be done in Matlab as follows. Make sure to open the olfactometer air valves and also run Clampex on the logging computer.

Initialize DAQ

The first step is to inform Matlab to perform data acquisition (DAQ). This is done by creating a data acquisition session:

```
airflowSession = daq.createSession('ni');
airFlowSession.IsContinous = true;
```

Where ‘ni’ is the vendor of the device, which for me is National Instruments. ‘isContinuous’ indicates that flow is a continuous variable. Now Matlab knows that there is a data acquisition session that involves a National instruments brand card.

Designate Channels

There are 3 MFCs, each connected to an analog output channel, as described in the Olfactometer Building Instructions section. But Matlab does not know this. The user must inform Matlab that these are the relevant channels:

```
addAnalogOutputChannel(airflowSession, 'Dev2', 'ao0', 'Voltage');
addAnalogOutputChannel(airflowSession, 'Dev2', 'ao1', 'Voltage');
addAnalogOutputChannel(airflowSession, 'Dev2', 'ao2', 'Voltage');
```

Where ‘Dev2’ is the device ID for my particular data acquisition card, ‘aoX’ is the name of the analog output channel into which each of my wires is physically plugged, and ‘Voltage’ indicates the type of analog signal to be sent along this channel.

Choose Flow Rates

The variable of interest is flow rate measured in mL/min. The Odor MFCs ranges are 0 – 100 mL/min and the Carrier MFC’s range is 0 – 1000 mL/min. Let’s choose the maximum flow rate for each:

```
F1 = 100;  % flow for odor 1
F2 = 100;  % flow for odor 2
F0 = 1000; % flow for the carrier stream
```

Convert Flow to Volts

The variable sent by Matlab is voltage, usually ranging from 0 – 10 V. The user must convert the flow rates to Volts using a simple linear relationship:

```
V1 = F1/10;
V2 = F2/10;
V0 = F0/100;
```

These relationships may vary, please check the MFC user manual.

Command the Flow Rates

Now, send each voltage to its corresponding MFC:

```
outputSingleScan(airflowSession,[V0,V1,V2]);
```

Since we defined ‘airflowSession’ to be a single session with 3 channels, the format for commanding each MFC is a 1x3 matrix in the order we specified. Once this is executed, the

olfactometer should produce a hissing sound. On Clampex, all MFCs command and feedback waveforms should have jumped from 0 V to 10 V. If the MFC feedbacks do not all read 10 V, then this is a problem. First, check the wiring. Next, check whether the equation from mL/min to Volts is correct. Next, make sure that you used the correct tubing width for the olfactometer. If the tubing is too thin, this creates a high-pressure path that can hinder MFC performance.

Try playing around with controlling different flow rates. You can also program functions to drive any waveforms of flow rates. Try a sinusoid:

```
frequency = 1; seconds = 10; tic;
while toc < seconds
     F = 45*(sin(2*pi*frequency*toc-pi/2)) + 55;
     outputSingleScan(airflowSession,[F/100,F/10,F/10]);
end
```

Each MFC should easily be able to produce these sinusoids. What are the frequency and amplitude limits of each MFC?

Notice that the voltage signals continue to be sent and the flows persists indefinitely. If left overnight or for days, this could cause wear and tear on the MFCs. It is the user's responsibility to take care of the system and install the proper failsafes. When the user is finished using the system, the final command should always be:

```
outputSingleScan(airflowSession,[0,0,0]);
```

This should shut off the MFCs, and the waveforms on Clampex should fall to 0 V. It is also a good idea to have all MFCs plugged into the same power strip that can be switched off at the end of the day.

Checking for Leaks

Next, the user must make certain that air is exiting the olfactometer only through the nose chamber. The simplest way to do this is to set all MFCs to maximum flow, fill a wide-mouthed container such as a large jar with water, and submerge each junction beyond the MFCs one at a time. If any junction produces bubbles, it must be dismantled and reconfigured. Teflon tape is useful to prevent leaks. This process involves dipping the air outlet ports of the MFCs slightly into water while they are running, so the user must proceed with care. This should not damage the device as long as the water does not touch the electrical ports. The user should regularly perform leak checks.

Choosing Odorants

The user should carefully consider the choice of odorant(s). An odorant should be:

1. A single chemical
2. Chemically stable in air
3. Moderately volatile

Single Chemical

Olfactory virtual reality experts (just me, I guess), frequently get asked questions like, “Why don’t you make it smell like cheese/peanut butter/pizza/beer/etc.?” There are many reasons why not. First of all, such smells consist of complex mixtures of tens or hundreds of different odorants, each with its own chemical stability and volatility. Remember that the goal of this system is to deliver odors consistently over long time periods to form reliable odor-spatial landscapes. Suppose we were to fill the olfactometer with liquid cheese. Certainly, a cheese odor could be delivered to the nose chamber. However, over minutes to hours, the different chemicals in the cheese would oxidize and vaporize at different rates, and the cheese would soon smell different from how it did at the start. During behavior in virtual reality, these inconsistent chemical

properties would lead to an unstable virtual cheese landscape, and thus the virtual reality would be a failure. Please do not fill the olfactometer with cheese.

Chemical Stability

Suppose we were to fill the odorant saturation chamber with a chemical that is unstable in air. This chemical could be delivered to the nose chamber. However, as time passes, the chemical would smell different than it did at the start. Thus, this odor would not be a reliable spatial feature in olfactory VR. The user must choose odorants that are stable in air. In practice, this generally means resistant to oxidation. Do not switch the olfactometer gas to nitrogen in an attempt to prevent oxidation, as this may cause suffocation.

Volatility

The odorant must begin as a liquid in the odorant saturation chamber, but end as a gas in the nose. Different liquids have different tendencies to vaporize into gases. This is commonly called volatility, and chemically quantified as vapor pressure. Since the olfactometer must run constantly for long durations, vapor pressure is a serious concern. A chemical with very high vapor pressure will deplete itself from the liquid source, and thus the smell will weaken over time. A chemical with low vapor pressure will fail to constantly saturate the head space of the odorant saturation chamber, and the smell will become very weak despite the liquid concentration remaining high. Thus, this system works best with odorants of "moderate" vapor pressure. The odorants methyl valerate (perceptually-optimal concentration 1:125 in mineral oil, vapor pressure 11.2 mmHg at 25 °C) and α-pinene (perceptually-optimal concentration 1:37.5 in mineral oil, vapor pressure 11.2 and 4.9 mmHg at 25 °C) have been validated to be reasonably stable in this system (Radvansky and Dombeck, 2018). However, odorant vapor pressures can range by orders of magnitude above and below these values. I have found the vapor pressure of Eugenol (0.009

mmHg at 25 °C) to be too low at perceptually-favorable concentrations (unpublished), and I suspect that the vapor pressure of methyl valerate may be near the upper limit at this concentration. Liquid odorant concentration can also play a role but the user should not stray too far from the perceptually-optimal concentration (see Choosing Odorants).

This is not to say that it is impossible to control odorants beyond this range, only that the it may be necessary to modify the olfactometer's configuration. For example, it is likely that the depletion issue of high vapor pressure odorants could be solved by using a larger bottle for the odorant saturation chamber, such that depletion becomes negligible. The saturation issue of low vapor pressure odorants could likely be solved by designing an the odorant saturation chamber to have a smaller head volume, by using a taller tube to maximize the surface area between the inlet air bubble and liquid odorant, or by limiting the range of that odorant's mass flow controller to only half its maximum. The user is responsible for any modifications made to the olfactometer, AND for validating such these modifications work. This can be done by following the procedure below.

Preparing an Odorant Saturation Chamber

As alluded to above, the design of the odorant saturation chamber is not trivial. These are instructions for preparing one configuration that can work for certain concentrations of certain odorants. It was chosen based on a series of human sniff tests and PID concentration measurements as outlined in the following sections. If this configuration does not work for the user's odorant, then at least it can serve as a reasonable starting point for future designs. To prepare an odorant saturation chamber of this design:

1. In the glass vials of the olfactometer, position the inlet tube to be deep into the vial, nearly touching the bottom. Position the outlet tube to be barely poking through the surface of the rubber membrane cap. The purpose of submerging the inlet tube is to

allow more time and area of interaction between the inlet air (bubbles) and the liquid odorant, to promote saturation of the head volume.

2. Fill 1 40-mL amber glass vial nearly to the top with glass beads, leaving ~1-1.5 cm of head space.

3. Fill 1 falcon tube with liquid odorant and mineral oil to a total concentration of 12 mL to make the odorant solution. For concentration, see the below section.

4. Shake the falcon tube vigorously using a vortex. It may take some effort to dissolve the solution.

5. Carefully pour the falcon tube into the glass vial. Sniff the glass vial and cap it. It is good practice to habitually sniff the odorant solution to confirm that they are correct, since there is no way to visualize them.

This filled odorant saturation chamber should be used on the same day that it is prepared. As discussed in the Chemical Stability section, odors can change over time due to oxidation, and thus mixed solutions should be discarded daily. Stock solutions should be capped with nitrogen gas each time they are closed. It is good practice to aliquot stock solution into many smaller airtight vials, and also nitrogen-cap them. Mineral oil is a good choice of solvent because it is very viscous and can reduce splashing when bubbling through the solution. It is also odorless, chemically stable, and unlikely to evaporate.

Choosing Concentrations

Since odor is a perceptual phenomenon that depends on receptors and neural circuits, there is no easy chemistry equation to choose what odorant concentration to use. The olfactory neuroscience literature may provide some reasonable starting points, but since this olfactometer is quite different from those used in previous experiments, the best practice is to first follow your nose, and second perform PID measurements.

The user should choose the perceptually-optimal odorant concentration as follows:

1. Prepare many odorant saturation chambers of concentrations ranging by order of magnitude, e.g. [100, 10, 1, 0.1, 0.01, 0.001]%;

2. Set the [carrier, odor1, odor2] MFC flow rates to [998, 1, 1] mL/min. These values are called the baseline. The MFCs should always be set to baseline when attaching or detaching an odorant saturation chamber. If an odorant saturation chamber is attached when the flows are off, this can cause the risk of backflow from the odorant saturation chamber to the MFC the instant that they are turned on. No MFC should ever fall below 1 mL/min when the odorant saturation chambers are attached.

3. Remove the empty glass vial from the odor1 pathway and replace it with the odorant saturation chamber of the lowest concentration. Screw it on tight to prevent leaks. Make sure that the beads do not touch the outlet tube. Once a particular odorant is chosen for a pathway, keep this designation for future sessions to avoid contamination.

4. Set the [carrier, odor1, odor2] MFCs to [899, 100, 1] mL/min. The final flow rate at the nose chamber should always be 1000 mL/min

5. Take a good sniff of the nose chamber, making sure to take all of it into your nostril(s). Take a few breaths of fresh air and try again. Take note of how strong or weak you perceive the odor to be. Keep in mind that this is the strongest possible smell that could be achieved from this configuration, without reducing the carrier stream flow rate, which is not recommended.

6. Set the MFCs to baseline. Remove the odor saturation chamber and wipe the inlet tube clean using kimwipes.

7. Attach the next chamber and repeat steps 3-6.

8. When finished, wipe the inlet clean with kimwipes, attach the empty vials, and set all MFCs to maximum flow for 30 minutes. This procedure is called “flushing,” and is to remove lingering odors. A good practice for flushing the MFCs is to write a script that automatically shuts them off after 30 minutes:

```
outputSingleScan(airflowSession,[10,10,10]);
pause(3600);
outputSingleScan(airflowSession,[0,0,0]);
```

The goal is to identify the order of magnitude of concentration that is distinctly perceptible but not “too strong.” A concentration that is too strong will lead to problems such as olfactory receptor saturation and olfactory adaptation, which could ultimately reduce the resolution of the olfacto-

spatial virtual landscape. Once an order of magnitude is identified, another round of sniff testing may be performed within that order of magnitude. If the subject is to be a non-human animal, then it is necessary to consult the olfactory literature for that species. As a rule of thumb, odor perception is for rodents is odors an order of magnitude more intense than for humans, and thus the user should choose a concentration that is barely perceptible at maximum flow rate.

Testing the Duration of Odorant Delivery

Once a perceptually-reasonable odorant concentration has been chosen, it is necessary to validate this concentration for long-duration use using the following procedure:

1. Prepare a Matlab script to drive the odor MFC with a sinusoid as described in Commanding the MFCs. The blank MFC should be set to 1 mL/min. At each timepoint, the carrier MFC should be set to F0 = 1000 – F1 – F2 to always produce a final flow rate of 1000 mL/min.

2. Prepare a Matlab script to drive the odor MFC with 20-minute square pulses of maximum amplitude, interspaced with 1-minute baselines of 1 mL/min including at the start and end. The blank MFC should be set to 1 mL/min. At each timepoint, the carrier MFC should be set to F0 = 1000 – F1 – F2 to always produce a final flow rate of 1000 mL/min. There should be enough 20-minute pulses to at least total the duration of your intended VR session. The purpose of the baselines is to correct for drift in the DC offset of the PID that can occur over these long times.

3. Set the MFCs to baseline ([998, 1 ,1] mL/min) with the vials empty and firmly fix the PID at the nose chamber.

4. Plug the PID output into the Digidata channel that is designated in the Clampex protocol. Turn the PID on and make sure that it is detected by Clampex. Set the PID gain to “Low,” the PID flow rate to 950 mL/min, and turn the DC offset to slightly above zero. The PID flow rate must be slightly lower than the oflactometer flow rate.

5. Replace an empty vial with the prepared odorant saturation chamber. Remember that the MFCs must always be set to baseline when attaching/detaching an odorant saturation chamber. Make sure that the inlet tube is at the bottom of the vial and the outlet tube is near the cap. Do not allow the beads and liquid to touch the outlet tube. Screw it on tightly.

6. Drive the odor MFC with a sinusoid and observe the waveform in Clampex. Reposition the PID until the waveform shows the highest signal-to-noise. Return the MFCs to baseline and firmly secure the PID.

7. Drive the odor MFC with the 20-minute square pulses until the script has finished. During this time, do not disturb the setup whatsoever.

8. Clean and flush the olfactometer as described in Choosing Odorants.

9. Load the Clampex data into Matlab or your preferred data analysis software. Take only the final 10 s of each baseline measurement and fit these points with a second-order polynomial (Figure 3).

10. Take only the non-baseline points and subtract the baseline polymonial. Fit the non-baseline points with an exponential (Figure 3). The time constant of this exponential is the time constant of the depletion of this odorant configuration at maximum flow rate. If this time constant is too short for the desired VR session, then a different odorant configuration must be chosen (return to Choosing Concentrations or re-design the odorant saturation chamber). If the time constant is acceptable, then this odorant configuration has passed the test and can be reliably delivered for long durations

11. Before testing the next odorant, replace the odorant saturation chamber with an empty vial, wipe the inlet tube, and flush the olfactometer for 30 minutes. This should remove residual odorant that could contaminate the next test. Designate one odorant per pathway and do not switch them.

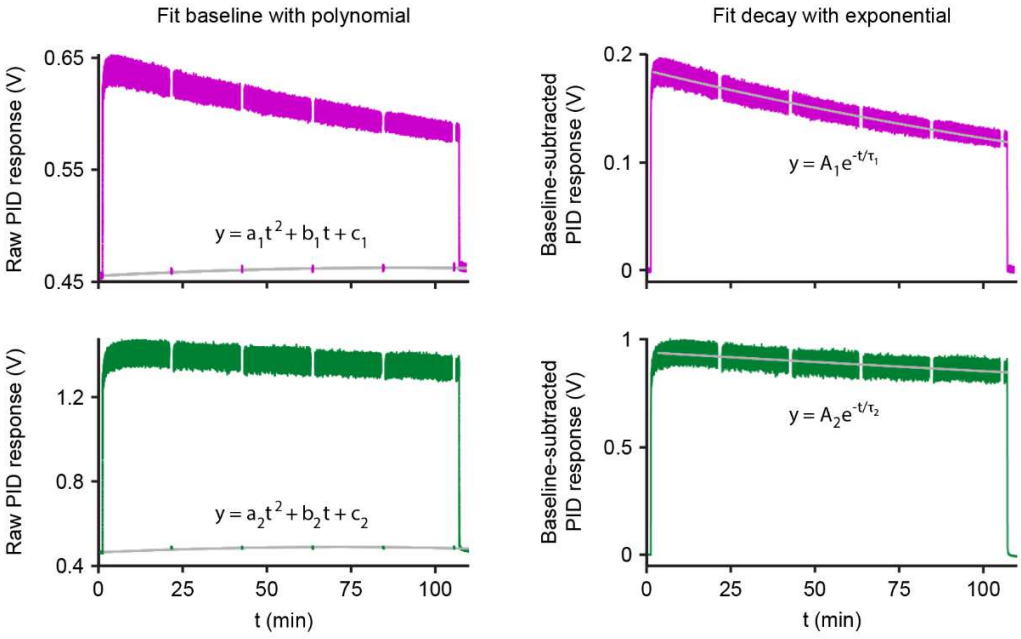

Figure 3. Procedure for calculating the time-constant of odorant depletion for 1:125 methyl valerate in mineral oil (top) and α-pinene 1:37.5 in mineral oil (bottom).

If your odorant is particularly "sticky," this can be a problem because the PID reading will decay as odorant residue accumulates on its sensor. I have found this to be the case with β-phenylethylamine. In this case, a trick can be performed. Fix the PID very rigidly and run the duration test. After each baseline and ~1 minute of each pulse, turn the PID off and carefully remove, clean, and replace the sensor. Turn the PID back on at the start of the next baseline. Repeat this for each pulse-baseline combination. In general, it is good practice to clean the PID from time to time (see its instruction manual).

When any change to the odorant configuration is made, this duration test must be repeated. Many factors can affect duration of odorant delivery, such as concentration, bottle size, whether or not the inlet tube is submerged, and whether or not the bottle is filled with glass beads to increase the surface area between the inlet air bubbles and the odorant liquid (Figure 4).

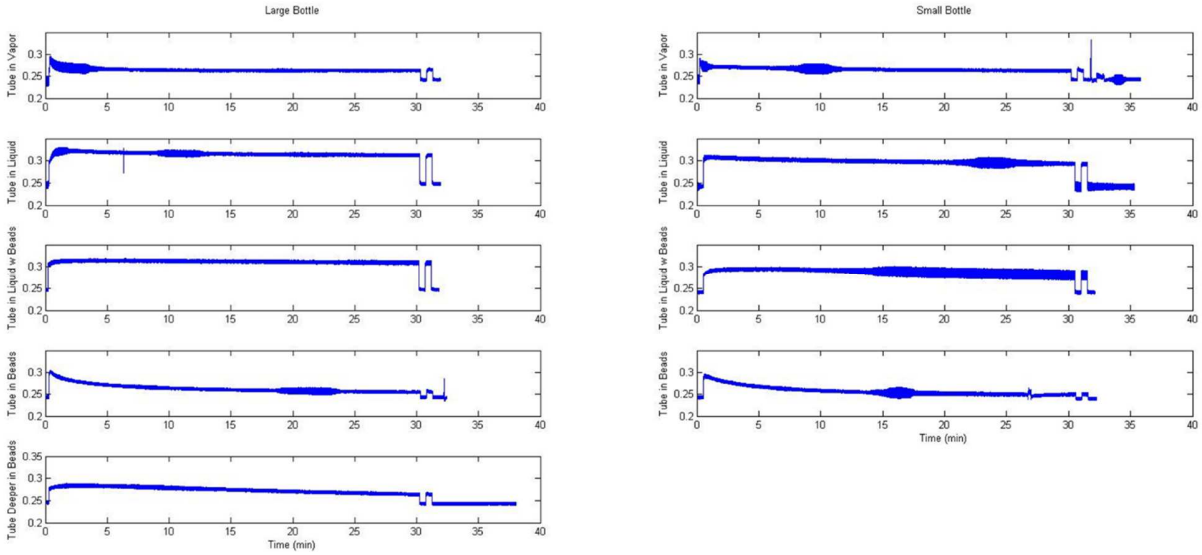

Figure 4. Duration of odorant delivery depends on bottle size, inlet tube immersion, and the presence of glass beads. Please ignore the fluctuations in noise level caused by artifacts within the PID, which should have been cleaned.

Testing the Speed of Odorant Delivery

Once a perceptually-optimal and stable odorant configuration has been chosen, the next step is to characterize its speed of delivery through the olfactometer. Because odorant delivery depends on the flow of physical molecules through tubes, there is an inherent mechanical delay between the time of MFC command and the time of odorant delivery at the nose chamber. This delay is problematic because it limits the olfacto-spatial resolution of the virtual environment during behavior, through which the subject should be free to move relatively quickly. Thus, this delay needs to be quantified, and ideally should be corrected for by an online algorithm (see Fine-Tuning the System). This is the procedure to quantify the delay of odorant delivery:

1. Prepare a Matlab script to drive the odor MFC with a sinusoid as described in Commanding the MFCs. The blank MFC should be set to 1 mL/min. Good parameter choices are peak-to-peak amplitude: 49.5 mL/min, frequency: 0.5 Hz, and DC offset: 25.75 or 75.25 mL/min, cycles: 150. It can be helpful to perform this test twice, one for each offset. Remember that the flow rates must always sum to 1000 and never fall below 1 mL/min.

2. Configure the olfactometer and PID as described in Testing the Duration of Odorant Delivery.

3. Run the script to drive the odor MFC with the sinusoid at low DC offset (Figure 5).

4. Run the script to drive the odor MFC with the sinusoid at high DC offset (Figure 5).

5. Clean and flush the olfactometer as described in Choosing Odorants.

6. Load the low-offset data from Clampex into Matlab or your preferred data analysis software. Calculate the lag between the odor MFC command sinusoid and the PID sinusoid. This can be done using cross-correlation (xcorr) or peak-to-peak subtraction (findpeaks).

7. Repeat this analysis for the high-offset data. Pool or average the delays from both offsets. This value is the delay of the olfactometer for this odorant in this configuration. Write it down. If this value is above ~200 ms, consider shortening the tube lengths.

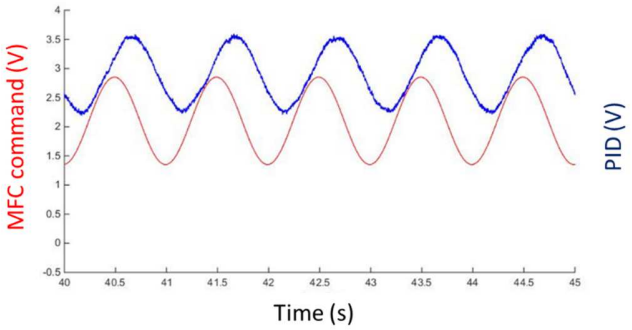

Figure 5. Odorant delivery lags MFC command.

If an odorant configuration has passed all of the above tests, then it is suitable for making and maintaining an olfactory landscape in VR. This procedure should then be repeated for the second odorant channel. To avoid cross-contamination and differences in delivery delay, odorants should always be restricted to their designated channels.

Implementing the System

Now that the user can rapidly and reliably control odorant concentrations as a function of time, these principles can be applied to control the odorant concentrations as a function of position in VR as driven by the VR controller.

Designing an Olfactory Virtual Landscape

First, the user must define the olfactory virtual landscape, that is, the distribution of odorant concentrations in space. A simple approach is to define the concentrations as linear gradients across a 1-dimensional spatial environment (Figure 6).

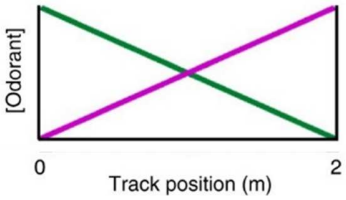

Figure 6. An olfactory virtual landscape defined by linear gradients.

The equations for these gradients are:

$$C_1 = \frac{100}{2}x \quad (1)$$

$$C_2 = 100 - \frac{100}{2}x \quad (2)$$

Where *C* is the normalized concentration measured in % and *x* is the position in VR measured in meters. Concentration can be normalized by defining 0% as the mean over several seconds of a minimum-flow (1 mL/min) PID reading, and 100% as the mean over several seconds of a maximum-flow (100 mL/min) PID reading. Alternatively, concentration could be measured in Volts (raw PID signal) or ppm (see the PID instruction manual). For these concentration gradients, the corresponding odor MFC flow rates *F* are:

$$F_1 = 1 + \frac{99}{2}x \quad (3)$$

$$F_2 = 100 - \frac{99}{2}x \quad (4)$$

Note that the relationship between concentration and flow is reasonably linear for this application in which the spatial frequency is quite low. For high-frequency applications, the flow equations may require nonlinear components (see Fine-Tuning the System). These spatial gradients are a simple example, but ultimately the user may choose any continuous equations for the olfactory virtual landscape, up to temporal frequencies of ~4 Hz, or spatial frequencies of ~8 m^{-1} for subject movement up to 0.5 m/s. It is also possible to include nonspatial components such as time and heading direction to simulate wind and nose orientation. Such components should be written into the equations at this stage. For validation purposes, deterministic equations are recommended such that the ideal olfactory landscapes can be exactly recreated from the logged data. If stochastic components are desired, consider randomly generating a library of stochastic landscapes, then selecting from these pre-determined landscapes during behavior (and logging each selection). An alternative is to log the ideal odorant concentration at each timepoint. For later validation, it must be possible to know exactly the ideal odorant concentration at all timepoints (see Fine-Tuning the System).

Designing a Task

Now that the environment has been defined, the controller can be implemented such that the subject can behave within the environment. A powerful and perhaps easy way to do this is by implementing an existing VR engine such as ViRMEn (Aronov and Tank, 2014), especially if using visual cues. The user should now decide between adding olfactory cues to an existing VR engine or building an olfactory VR engine from the ground up. For the sake of completeness and

independence from other methods, this manual will describe the latter approach, though the same principles apply to both approaches.

It is useful to think of the VR engine in three sequential sections: **Initialization**, **Runtime**, and **Termination** (see also the ViRMEn instruction manual). **Initialization** runs once, and defines the task variables such as the odor landscape equations and the gain to transform the controller signal into movement in VR. **Runtime** runs indefinitely, and each iteration inputs the controller signal and updates the task variables such as position and MFC flow rates. **Termination** runs once, and sets the task variables such as MFC flow rate back to baseline. It is good practice to divide the VR engine code into the above three sections.

The runtime function requires a loop that runs until the session is terminated. One way to do this in Matlab is to make a button using a graphical user interface (GUI).

```
%% Initialization
ButtonHandle = uicontrol('Style', 'PushButton', ...
                         'String', 'Stop VR',
                    'Callback','delete(gcbf)';
tic

%% Runtime
while(1)
    pause(1e-100); % pause briefly to update the figure
    if ~ishandle(ButtonHandle) % if the stop button is pressed
        break; % break the loop
    end
end

%% Termination
toc
```

Such a GUI can be a good place to monitor task variables online, in addition to Clampex.

Generating an Olfactory Virtual Landscape

After defining the Equations (3,4) for the olfactory virtual landscape, the user can program these equations into the VR engine. In the **Initialization**, set up the data acquisition session as described in Commanding the MFCs:

```
airflowSession = daq.createSession('ni');
airFlowSession.IsContinous = true;
addAnalogOutputChannel(airflowSession, 'Dev2', 'ao0', 'Voltage');
addAnalogOutputChannel(airflowSession, 'Dev2', 'ao1', 'Voltage');
addAnalogOutputChannel(airflowSession, 'Dev2', 'ao2', 'Voltage');
```

The equations/parameters can also be defined here if desired. Next, implement Equations 3 and 4 in the **Runtime**:

```
F1 = 1   + x*99/2;
F2 = 100 - x*99/2;
F0 = 1000 - F1 - F2;
V1 = F1/10; V2 = F2/10; V0 = F0/100;
outputSingleScan(airflowSession,[V0,V1,V2]);
```

Where *x* is position in VR measured in meters ranging from 0 to 2. Of course, the position could be defined in any arbitrary units, but it is simplest to scale everything directly to the real circumference of the ball (see next section). Position has not been defined yet, so this program will not run. A simple trick for testing the system without a controller is to define position based on the clock. Try this statement at the start of the **Runtime**:

```
x = mod(toc,2);
```

And this statement in the **Termination**:

```
outputSingleScan(airflowSession,[1/10, 1/10, 998/100]);
```

The MFCs should cycle between 1 and 100 mL/min every 2 seconds until the session is terminated.

Reading from the Controller

Instead of using the clock, *x* can be defined using the VR controller. The controller has already been wired to the VR computer and to the logging computer (see Virtual Reality Controller). You should be able to spin the ball and observe the optical mouse reading in Clampex on the logging computer. The next step is to read this signal into Matlab in the VR computer. This can be done by initializing a session (see Commanding the MFCs) for the controller in the **Initialization**:

```
mouseReadSession = daq.createSession('ni');
```

Where 'ni' indicates the vendor of the data acquisition card, National Instruments. Next, designate the channels for each dimension of the optical mouse treadmill reader:

```
addAnalogInputChannel(mouseReadSession, 'Dev1', 'ai0', 'Voltage');
addAnalogInputChannel(mouseReadSession, 'Dev1', 'ai1', 'Voltage');
```

Where 'Dev1' is the device ID for my particular data acquisition card, 'aiX' is the name of the analog input channel into which each of my wires is physically plugged, and 'Voltage' indicates the type of analog signal to be read from this channel. To read from this channel at each iteration of the **Runtime**:

```
moveData = inputSingleScan(mouseReadSession);
```

This creates a 1x2 array called 'moveData' containing the voltages readings from each dimension of the optical mouse. Note that 'inputSingleScan' performs an instantaneous reading that can lead to 'jerky' movements due to imperfections or wobbling of the Styrofoam ball. A better approach is to smooth these signals by averaging the previous several optical mouse readings. This will be

left as an exercise for the user (see also the ViRMEn code and instruction manual). For 1-dimensional treadmills, rotary encoders tend to be quite accurate and may not require smoothing.

Logging the Experiment Variables

So far, the signals sent and received by the VR computer are logged in Clampex. But some signals are actually generated by the VR computer and not sent anywhere, such as position and heading in VR. These variables need to be logged in Clampex as well. To do this:

1. Wire an analog output channel of the data acquisition card to an analog input channel of the Digidata. This channel is for position.

2. Repeat step 1 for the heading channel.

3. In the **Initialization**, designate these channels for position and heading, and set an initial value for heading *theta*:

```
vrLogSession = daq.createSession('ni');
addAnalogOutputChannel(vrLogSession, 'Dev1','ao2','Voltage');
addAnalogOutputChannel(vrLogSession, 'Dev1','ao3','Voltage');
theta = 0;
```

   theta is the heading in the horizontal plane, measured in radians. The user is free to choose which heading corresponds to "north." Decide this and write it down. Here, north is $\pi/2$, so *x* runs east-west.

4. Since position and heading are measured in meters and radians, these values need to be converted to Volts to be sent to Clampex. Write out the equations to linearly scale position and heading to Volts, within the dynamic range of the hardware (usually $\pm$ 10 V). In this case, position ranges from 0 to 2 m and heading ranges from -pi to pi rad. In the **Runtime**, write these equations:

```
V1 = -9 + (18/2)*x;
V2 = (mod(theta+pi,2*pi) - pi)*18;
```

5. In the **Runtime**, output these voltages to Clampex:

```
outputSingleScan(vr. vrLogSession, [V1,V2]);
```

6. Make sure that the Clampex protocol is set to read these channels.

Now the position and heading can be monitored and logged.

Moving through the Olfactory Virtual Landscape

Now that the optical mouse signal is in a Matlab variable, it can be transformed into movement in VR. This requires two steps. First, multiplying the optical mouse signal by constant gain factors to convert Volts into meters or radians. Second, performing geometry to calculate the velocity vector in VR. The gain factors must be determined empirically. To calculate the pitch (forward-moving) gain factor *g1*:

1. Measure the circumference *c* of the treadmill in meters. For each pitch revolution, the subject should advance in VR by *c* meters.

2. Calculate how many pitch revolutions are required to traverse the track (which in this case is 2 m long).

3. Mark the ball with tape or a pen such that revolutions can be counted.

4. In the **Initialization**, set *x = 0*. Set the pitch gain *g1 = 10*. Run VR.

5. In the **Initialization**, start a clock using *tic*.

6. In the **Runtime**, observe the elapsed time as *dt = toc*; and start a new clock with *tic*;. Since the *toc* of the very first iteration does not represent an iteration in VR, include an *if* statement to force *dt = 0* on the very first iteration.

7. In the **Runtime**, set *x = x + g1*moveData(1)*dt*. This is simply a first-order kinematic equation. The user may optionally include second-order kinematics or other physics here.

8. In the **Runtime**, set the condition *if x < 0; x = 0;* end; This creates an impassable "wall" at the start of the track.

9. In the **Runtime**, set the condition *if x > 2; x = 2;* end; This creates an impassable "wall" at the end of the track.

10. Start VR.

11. Carefully rotate the treadmill in pitch only by the number of pitch revolutions for one track traversal. Monitor the distance traversed by observing the position in Clampex.

12. If the track end was not reached, increase *g1*. If the track end was reached too early, decrease *g1*. Repeat this process until the gain is exactly correct. Repeat this several times to account for human error in rolling the treadmill.

Now the subject can traverse the olfactory virtual landscape by moving the treadmill. But currently, the only way to traverse the landscape back again is to spin the treadmill backward. This is not very useful. If the user intends to use a 1-dimensional treadmill (or only 1 dimension of a 2-dimensional treadmill as described thus far), then some easy tricks can be applied to get the subject "unstuck" from the track end. The first trick is to "teleport" the subject back to the start. To do this, simply impose the following condition in the **Runtime**: *if x > 1.99; x = 0; end;* This method will cause abrupt jumps in the odorant concentration, and is thus not always ideal for creating continuous odor landscapes. The second trick is to turn the subject around by 180^{o}. In the **Runtime**: *if x > 1.99; g1 = -10; end;* and *if x < 0.01; g1 = +10;* end; (or whatever your gain value is). This can be preferable to teleportation because maintains a smooth odor landscape. Note that the subject would not necessarily perceive this event as a turn-around, but rather as a lap along an infinitely long track in which the two odors oscillate as spatial triangle waves out of phase.

If instead the subject is to use the controller to turn around manually, then the yaw velocity signal must be incorporated. First, its gain *g2* must be calibrated. This process is more intuitive when visual cues are present, but can still be done using the same process used to calibrate *g1*. To do this, first decide how many yaw revolutions of the treadmill should be required to make one revolution in VR. There is no obvious choice for this value; it depends on the types of movements made by the subject. A high value can cause the subject to spin around wildly with only a small movement; a low value requires the subject to expend great effort to turn. A reasonable starting value is 5 treadmill revolutions per VR revolution. Once a value is chosen, calibrate the yaw (angular) gain *g2* using the same procedure as for *g1* above.

Now that the pitch and yaw (forward and angular) gains are determined, they can be integrated to calculate the velocity vector in VR. This can be done using basic trigonometry:

```
x = x + g1*moveData(1)*cos(theta)*dt;
y = y + g1*moveData(1)*(-sin(theta))*dt;
theta = theta + g2*moveData(2);
```

Make sure that these equations are correct for your northward convention. Since this is a 1-dimensional track, the y component of the vector is not used, and can be commented out. The subject can now move back and forth and turn around along the linear track.

Generalizing to Two Dimensions

The system can now be scaled up for 2-dimensional applications, if desired. Consider a 2x2 m arena with odorant concentrations decaying radially from point sources (Figure 7).

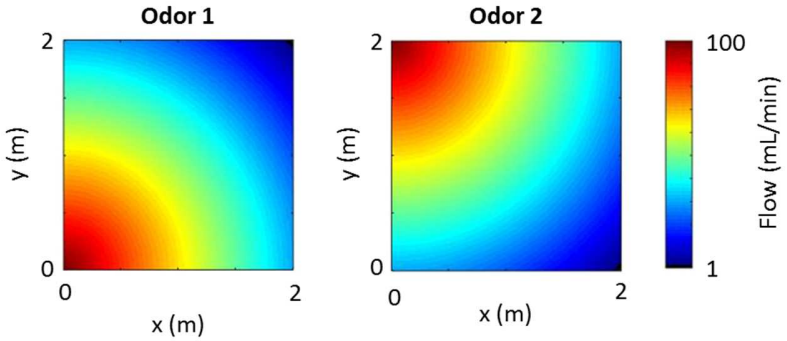

Figure 7. An olfactory virtual landscape defined by radial concentration gradients.

The equations for this landscape are:

$$F_1 = 100 - \frac{99}{2} r_1 \quad (5)$$

$$F_2 = 100 - \frac{99}{2} r_2 \quad (6)$$

Where r is the radius from the corresponding odor source.

To create this arena, modify the linear track as follows:

1. To log the y-position, connect an analog output channel from the data acquisition card to an analog input channel of the Digidata. Include this channel in the Clampex protocol. Assign this channel in the VR code and scale meters to Volts as described above for *x*.

2. In the **Initialization**, define *y = 0*; (or whatever start location you prefer).

3. In the **Runtime**, un-comment the y kinematic equation. Create "walls" at *y = 0* and *y = 2* as described above for *x*.

4. In the **Runtime**, calculate r_1 and r_2, and replace Equations 4,5 with Equations 5,6.

Now the user can move through a 2-dimensional odor landscape in VR. Note that 2-dimensional VR is less established than 1-dimensional VR, and the neural mechanisms engaged can vary depending on the subject's behavior (Aronov and Tank, 2014; Aghajan et al., 2015). The fact that a virtual landscape exists does not necessarily mean that the subject uses it for meaningful spatial behaviors. The burden is on the user to perform a behavioral experiment to validate this.

Adding Non-Olfactory Cues

Building a visual virtual environment can be rather difficult. Fortunately, many visual VR systems already exist, so there is no need to reinvent the wheel. The best way to incorporate visual cues is to begin with an existing visual VR engine and perform the above procedures to include odor cues. The ViRMEn engine (Aronov and Tank, 2014) is quite powerful and user-friendly, and also freely available from pni.princeton.edu. This engine runs in Matlab and is organized by the sections described above: **Initiation**, **Runtime**, and **Termination**, and thus the above code can be written directly into the corresponding sections of ViRMEn. Note that ViRMEn uses a separate section where the movement function should be written. Visual cues can be hand-drawn by following the ViRMEn instruction manual. Obviously, this will require additional components such as monitors/projectors to be built into the apparatus.

There do exist tactile (Sofroniew et al., 2014) and auditory (Funamizu et al., 2016) VR systems that could potentially be interfaced with the Smellevision. However, these systems require significant engineering beyond the scope of this manual. Simpler stimuli such as auditory tones, mechanized tactile pads, and reward delivery systems are relatively easy to incorporate. Simply wire the device to an analog output from the data acquisition card, designate the channel in Matlab, and send voltages to the device triggered by the subject's behavior, such as occupying a certain position zone. Likewise, input devices such as push-buttons or lick sensors are straightforward to incorporate. Wire the device to an analog input channel, read it into Matlab, and trigger events in VR when the device is activated and the proper conditions are met (e.g. "to get a reward, press the button when you have found the odor source location.").

Validating the System for Behavior

Now that the Smellevision has been built, behavioral sessions can be performed. It is not safe to assume that the odor landscape is perceived or that it has any behavioral relevance. In fact, behavior and neural responses can depend heavily on whether the subject pays attention to the odors (see Chapters 3 and 4 of this dissertation). The user must design a behavioral task that depends on perceiving the odor landscape, and demonstrate that the subject can perform this task. An example task could be to navigate to a randomly-generated odor target, then stop or press a button once the odor target is found. The user must control for all possible confounds such as navigating based on other sensory cues, as well as by using intrinsic strategies such as timing or step-counting. Once this behavior is apparently established, the user should take the data from Clampex and analyze them methodically to rule out any confounds.

Fine-Tuning the System

After establishing an odor-guided behavioral paradigm, the user must validate that the supposed olfactory landscape is maintained during this behavior. If it is not, then corrections can be applied. This is done after establishing the behavior because the fidelity of the landscape depends on the subject's movement. The best way to do this as follows:

1. Create a "standard" behavioral session. This is done by taking a good behavioral session from Clampex and downsampling it to roughly the refresh rate of the VR engine.

2. In the VR engine, load the standard behavioral session as a variable in the **Initialization**.

3. In the **Runtime**, instead of defining position and heading by reading the controller, define them by the standard behavioral session at the corresponding timepoint.

4. Set the olfactometer to baseline and attach one odorant saturation chamber.

5. Set up the PID firmly at the nose chamber as described in Testing the Duration of Odorant Delivery.

6. Take PID readings for ~10 s at baseline (1 mL/min) and maximum (100 mL/min) odor MFC flow rate. These are used later to normalize the PID signal.

7. Run the VR. In Clampex, you should see the standard behavior replay in real-time.

8. When the replay is complete, load the Clampex data into Matlab or your preferred analysis software.

9. Normalize the PID signal to the mean of the baseline and the mean of the maximum to get odorant concentration.

10. Plot the ideal odorant distribution Equation 1 (or 2). Overlay the real odorant distribution. For the linear track in Figure 6, this is the result:

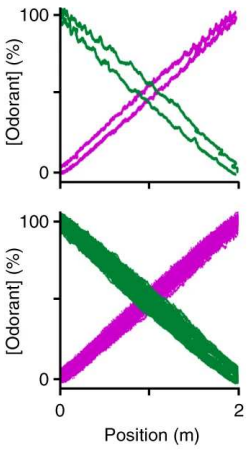

Figure 8: Real odorant spatial distribution for 2 traversals in opposite directions (top) and many traversals in both directions (bottom)

The ideal "X" (Figure 6) is somewhat reproduced, but there is some error (Figure 8). This is due to the delay in the olfactometer (see Testing the Speed of Odorant Delivery). As the subject runs "rightward," the X is skewed to the left. As the subject runs "leftward," the X is skewed to the right.

11. There are two options here. a.) Quantify this error and accept it as a limit to the odor-spatial resolution. b.) Correct this error using a predictive algorithm.

Position-Predictive Algorithm

The predictive algorithm is conceptually straightforward. Since there is a delay *d* between commanding the odor MFC and receiving the odorant at the nose chamber (see Testing the Speed of Odorant Delivery), the odor will always lag behind the behavior. To correct for this, the user can instead drive the MFCs based on the predicted future behavior. In this case, the behavioral variable of interest is position *x* (though this process could be applied to any number of behavioral variables such as heading *theta* and position *y*). To apply the position-predictive algorithm:

1. Write out the equation $x_f = x_0 + v_o d$, where x_f is predicted future position, x_0 is current position, v_0 is instantaneous velocity, and *d* is olfactometer delay for this odorant configuration. All of these independent variables are known except v_0, which depends on the number of previous VR iterations that are considered to be "instantaneous." Considering only one previous iteration may be error-prone, and can cause the algorithm to wildly overshoot and undershoot the true future position. Considering too many iterations will "oversmooth" the prediction and average out fast changes in movement.

2. To determine the optimal number of previous iterations for calculating v_0 (and ultimately calculating x_f), load the downsampled standard behavior in Matlab. Create a "pseudo-VR" engine consisting of **Initialization**, **Runtime**, and **Termination** sections.

3. In the **Runtime**, loop each timepoint of the standard behavior. At each timepoint t, calculate x_f with v_0 defined as $\Delta x/\Delta t$ since the 1 previous iteration. Calculate the error between your calculation of x_f and the true future position at timepoint $t + d$. Calculate the absolute sum of error over all timepoints. For this and subsequent error calculations, it is acceptable to discard timepoints when the subject is not moving (and thus the predictive algorithm is not expected to have any effect).

4. Repeat Step 3 using a time window of 2 iterations, i.e. defining v_0 as the average of the previous 2 $\Delta x/\Delta t$ measurements. Repeat this for time windows up to ~40 iterations.

5. Plot the absolute sum error as a function of time window (Figure 9).

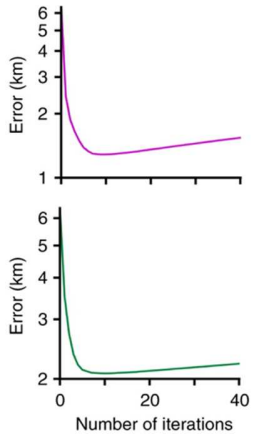

Figure 9. Optimization of time window for calculating instantaneous velocity.

The window with the minimum error is the optimum.

6. Program the predictive algorithm into the real VR engine. This will require a buffer variable to store the previous several Δx and Δt values.

7. Repeat the replay PID test with the predictive algorithm on vs. off, keeping the PID firmly fixed in place between the two tests (though refreshing the odorant saturation chambers). Overlay plots of the real and ideal odorant spatial distribution (Figure 10).

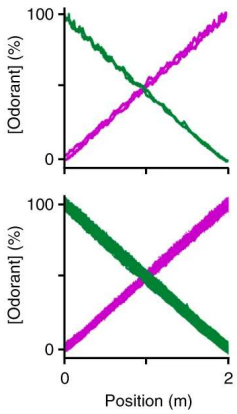

Figure 10. Real odorant spatial distribution with a position-predictive algorithm for 2 traversals in opposite directions (top) and many traversals in both directions (bottom)

8. Calculate the improvement provided by the algorithm as the error between the ideal and corrected odorant spatial distribution divided by the error between the ideal and uncorrected odorant spatial distribution.

Position-Amplitude-Predictive Algorithm

For generating high-frequency olfactory landscapes, the position-predictive algorithm alone may not be sufficient to correct for errors. For example (Figure 11):

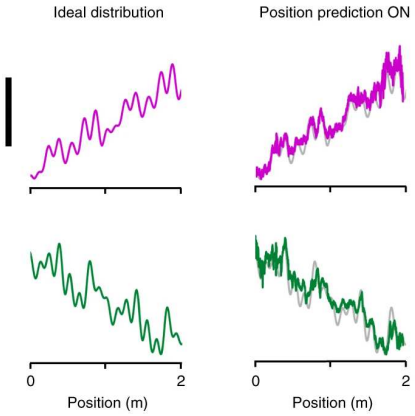

Figure 11. Position-predictive algorithm fails to correct errors in a high-frequency olfactory landscape. Scale bar 30%.

The high-frequency waveform is not reproduced. This is due to a variable relationship between the rate of change of MFC flow rate and the resulting odorant concentration change detected at the nose chamber one odorant mechanical delay in the future, and a variable odorant mechanical delay

as a function of control drive frequency. This can be corrected by including an amplitude component to the predictive algorithm. This component works by exaggerating the odorant stream flow rates by a magnitude that is a function of the desired concentration change (difference between current concentration and desired concentration one odorant mechanical delay in the future). This position-amplitude-predictive algorithm can be created using the following procedure (Figure 12, Figure 13):

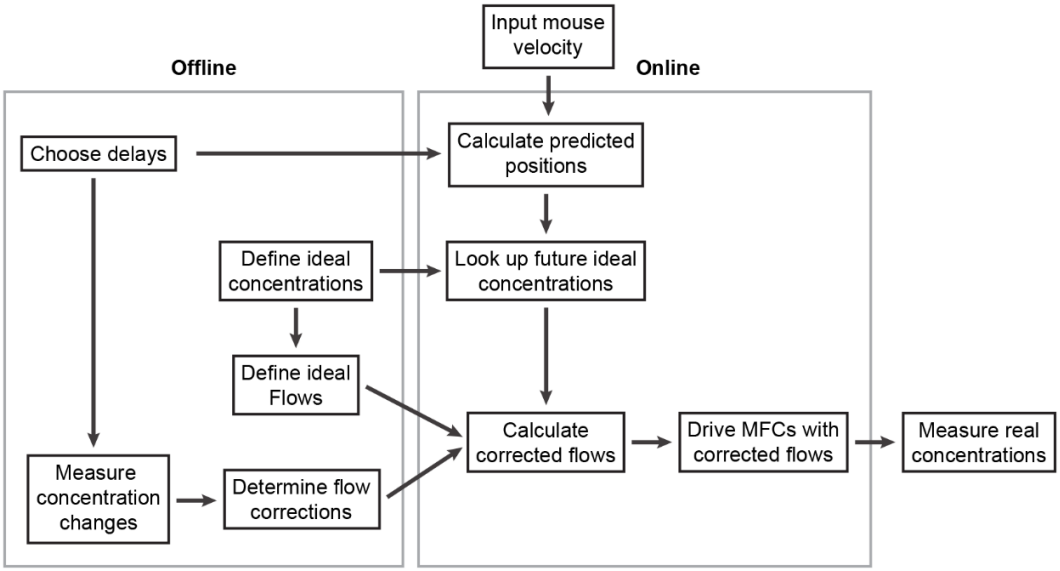

Figure 12. Block diagram for the position-amplitude-predictive algorithm.

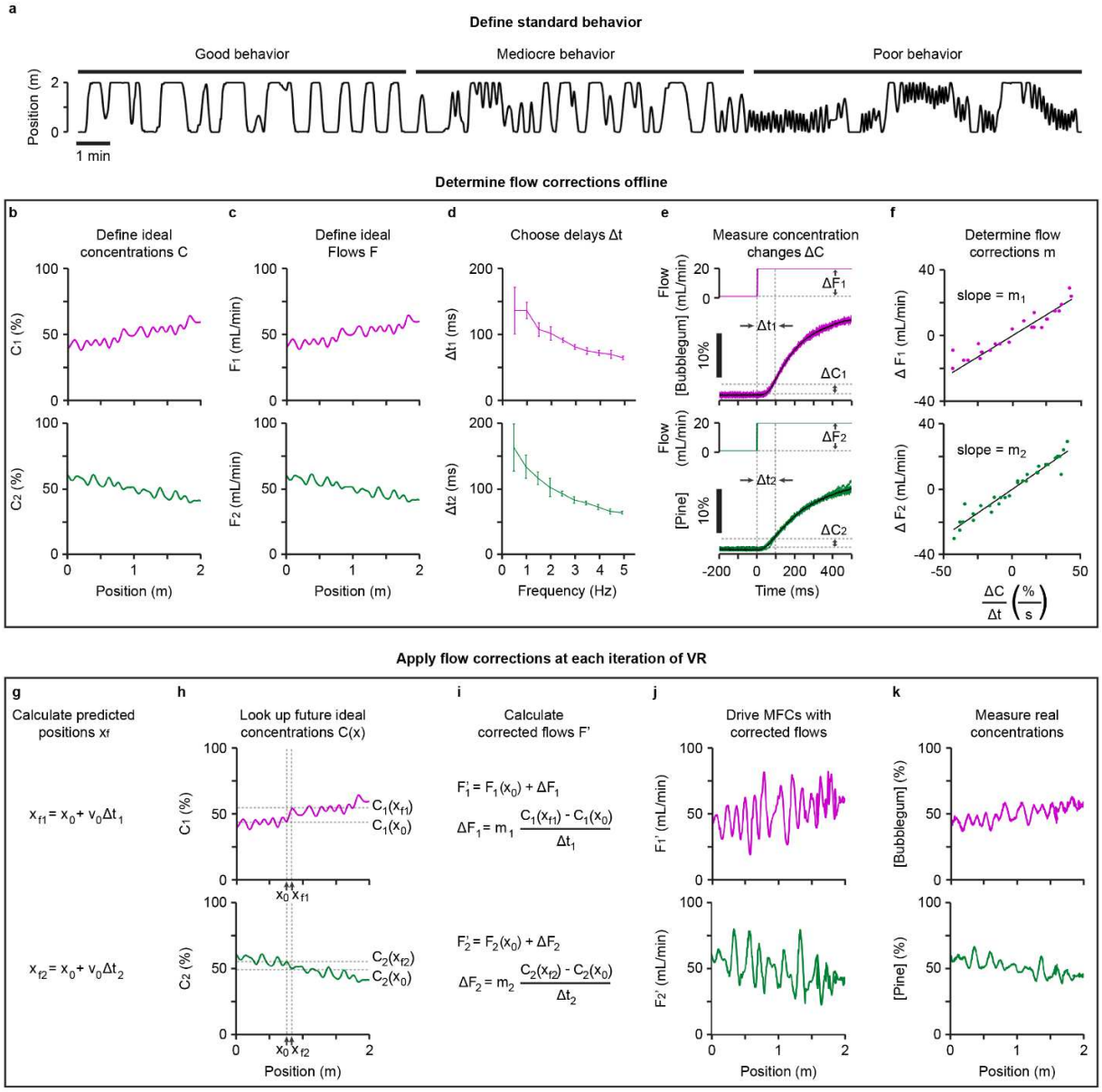

Figure 13. Step-by-step outline of the position-amplitude-predictive algorithm

a) Define a standard testing set as described in Position-Predictive Algorithm

b) Define the ideal concentration C_i distributions to be created.

c) Define the ideal flows as $10*C_i$. These are the flows that would produce the ideal concentration distributions in the absence of any mechanical constraints. Note that the odor flow rates must never fall below 1 mL/min, including overshoots introduced by this algorithm. It is recommended to raise the baseline to ~30 mL/min.

d) Choose the delays that best represent the frequency components of the ideal concentration distributions. These plots were made by applying 10 cycles each of sinusoids of frequencies [0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0,4.5, 5.0] Hz and amplitudes [10, 20, 30, 40, 50, 60, 70, 80, 90, 99] mL/min, and calculating the mean peak-to-peak delay between the MFC command and the PID signal. Mean delays tend to be relatively amplitude-invariant, and thus can be pooled over all amplitudes.

e) Measure the change in concentration that occurs Δt after a change in flow of a given amplitude. Concentration plots show 10 triggered signals (color) overlaid with their averages (black).

f) From the data in **e**, plot change in flow as a function of the slope of the concentration change. These plots include flow steps upward from 1 mL/min, downward from 100 mL/min, and outward from 50 mL/min. Beyond the ranges shown, the plots can become non-linear, do not have to be used for this algorithm unless points occur within this range during behavior. The slope m of the best-fit line with y-intercept forced to be 0 is the amount of flow change needed to produce a given concentration change by time Δt. For some applications, this linear fit results in an overshoot of the ideal concentrations. This overshoot can be counteracted by instead fitting the points with a logistic function, thus including a saturation component to the change in flow rate.

g) Use the subject's initial position and velocity to predict its future position at each Δt in the future, as described in Position-Predictive Algorithm.

h) Determine the concentration at each future position by looking up the values from the ideal concentration distributions shown in b.

i) Use the m-values from f to calculate the flows needed to reach the ideal concentrations at Δt in the future.

j) Drive the MFCs with the flow rates determined in **i**. These plots are the command signals to the MFCs for one traversal of replayed behavior through the ideal concentration distributions shown in **b**.

k) Measure the resulting concentrations at the nose chamber. These are the PID signals during the traversal replayed in **j**.

l) Overlay plots of the real and ideal odorant spatial distributions (Figure 14).

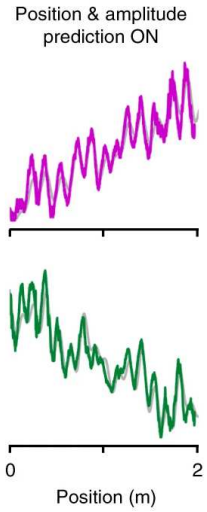

Figure 14. Overlaid plots of the real and ideal odorant spatial distributions.

m) Plot the power spectrum, error, and cross-correlation of the real vs. ideal odorant spatial distributions to ensure that the amplitude and temporal components are comparable.

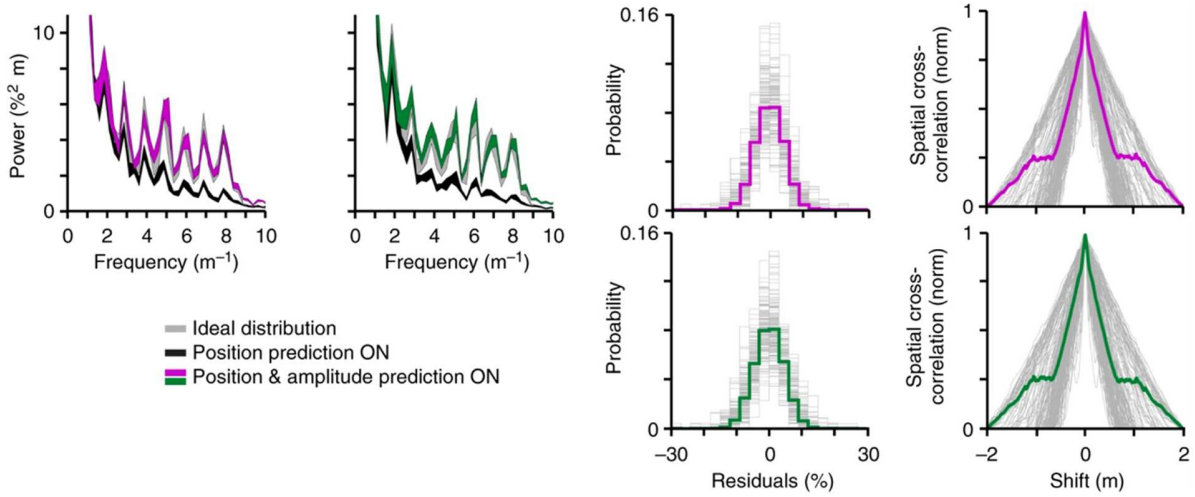

Figure 15. Comparing the ideal odorant spatial distribution with the position-amplitude-corrected odorant spatial distribution.

This position-amplitude-predictive algorithm is complicated, but as shown in Figures 14 and 15, it can greatly enhance the system's performance for high-frequency applications. This algorithm can also be used to control odor "plumes," that are likely to be more realistic and relevant for real-world navigation than the linear gradients used in the above examples (Figure 16).

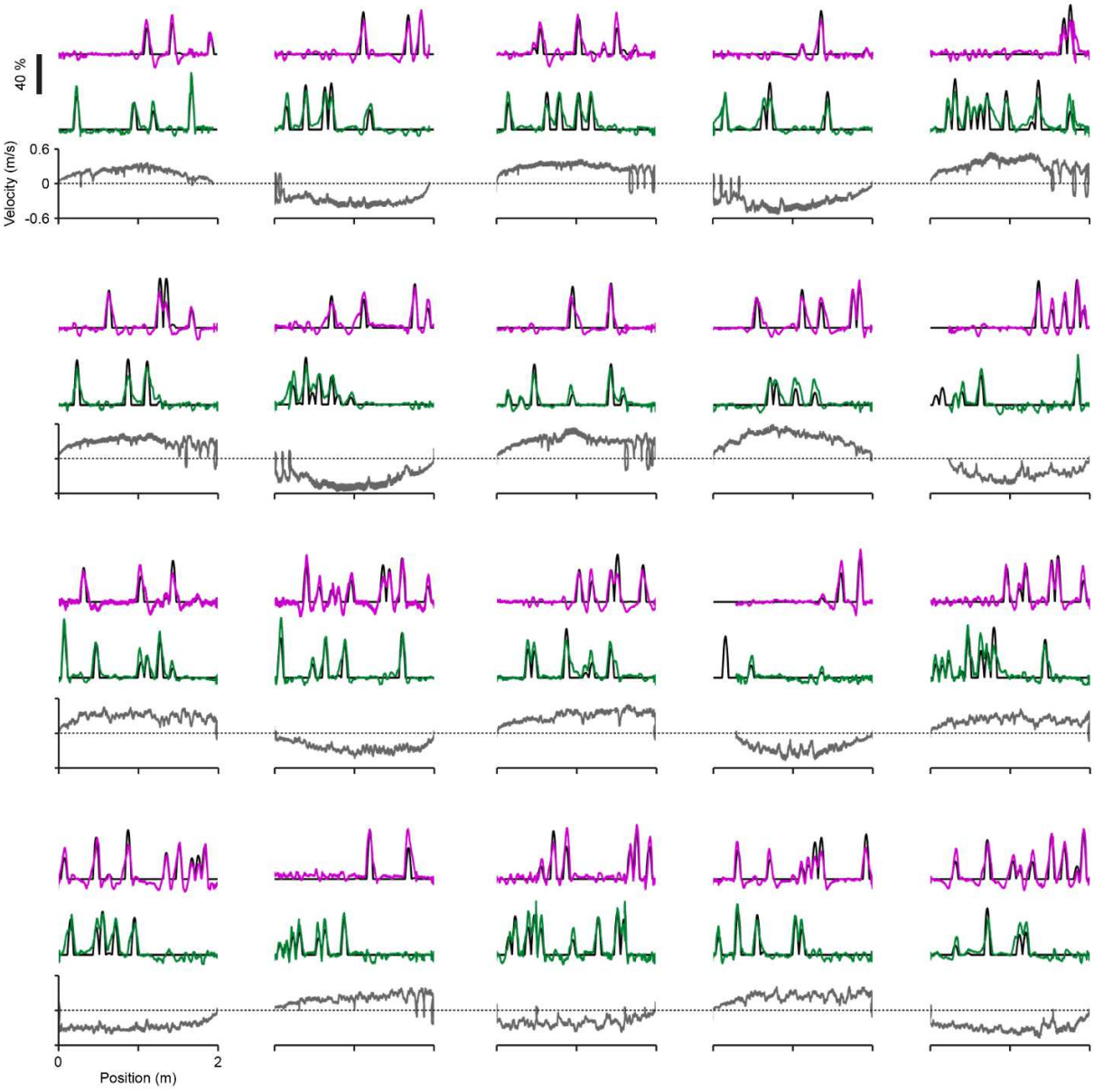

Figure 16. The position-amplitude-predictive algorithm can control odor plumes during behavior. Each panel is one traversal along a linear track. Black: ideal odorant spatial distribution, color: real odorant spatial distribution, gray: mouse running velocity.

Upkeep

Once the Smellevision has been built and validated by all of the above tests, it can be used to rapidly and reliably control odorant concentrations as spatial variables in VR. But as with any machine, it can break over time. The user must regularly perform the following upkeep steps to ensure that the system is working.

Checks to Perform Each Time the System is Used

1. After preparing the odorant saturation chambers, smell them. If not stored properly, odorant solutions can oxidize and their smells can change. If the odor smells different from the fresh solution, new odorant bottles must be prepared from the stock. It is a good idea to have several fresh stock bottles available in case the stock has also oxidized. It is good practice to aliquot the stock into many small airtight bottles, gently blow nitrogen into each one, then quickly screw on the cap and wrap it in parafilm.

2. Check the air filter and make sure it is not depleted. If so, replace it. It is good practice to have several air filters in stock.

3. Make sure that the air valves are all open. It is fine to leave the air valves open except when changing the filter or dismantling the olfactometer. Running the MFCs without any inlet air can damage them.

4. Attach the odorant saturation chambers and run VR. Always set the MFCs to baseline before attaching/detaching the vials.

5. Observe the tubes between the MFCs and the odorant saturation chambers. Make sure there is no backflow of odorant/mineral oil. If there is, the MFCs must be cleaned with isopropanol or sent to the manufacturer for repair (see the MFC user manual).

6. Make sure that the controller is reading properly. Give it a good spin in multiple directions and see whether the reading on Clampex is smooth. The optical mouse should always be as close to the ball as possible without touching. If this distance changes, then the gain factors for calculating velocity will be incorrect, and movement in VR will be inaccurate.

7. Make sure that each MFC feedback in Clampex is smooth.

8. Place your nose at the nose chamber and perform the odor-guided task yourself. Make sure that you can sense all odors at the proper intensities.

9. Make sure that all other task-related devices are working during VR. Reward delivery systems, for example, can dry out overnight.

10. After finishing VR sessions for the day, remove the odorant saturation chambers, wipe the inlet tube with kimwipes, and replace the chambers with empty vials. Flush the olfactometer for 30 minutes of maximum flow, then turn all MFCs off.

11. Note that the soda lime glass beads can be reused if they are washed thoroughly with isopropanol and dried completely. A fan can be helpful for air-drying. Before reusing the beads, smell them to make sure that they do not smell like odorant or isopropanol.

Checks to Perform Periodically

1. Leak check. With the odorant vials empty and the MFCs at maximum flow, submerge each tubing junction beyond the MFCs in a wide jar of water and make sure that there are no bubbles. This will require slightly dunking the MFC outlet port into water. This is fine as long as the electric outlets do not touch the water.

2. Speed check. Measure the speed of each odorant delivery as described in Testing the Speed of Odorant Delivery.

3. Replay check. Replay the standard behavior with the PID attached as described in Fine-Tuning the System. Make sure that the ideal and real odorant spatial distributions match.

4. Alcohol flush. If flushing the MFCs with air is insufficient to remove residual odors, the olfactometer can be cleaned by flushing with a small amount of isopropanol in the vials. This is generally not necessary except when switching to a new odorant.

Please take good care of your Smellevision! Thank you for reading this manual, and good luck.