University of Nebraska – Lincoln

Department of Psychology

Canine Cognition and Human Interaction Laboratory

Dr. Jeffrey R. Stevens

Precise Dispenser for Canine Numerical Discrimination Tasks

7 March 2021

V0.1

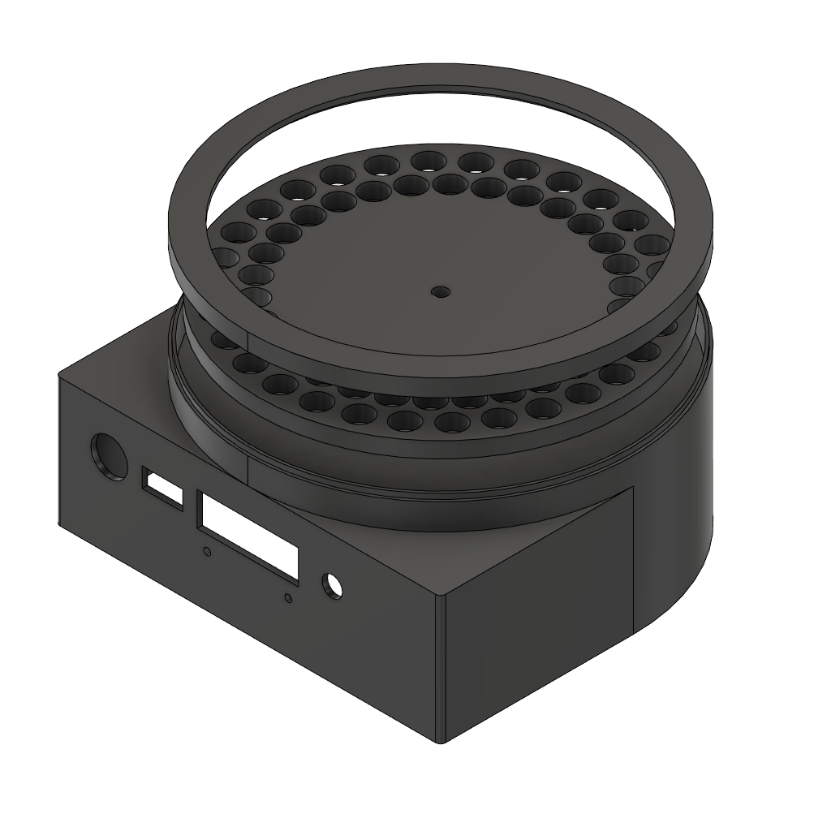


Figure 1 – Dispenser v33 Render

**Overview**

The performance of canine numerical discrimination tasks can be hindered by a lack of appropriate instrumentation. Using manual, researcher-controlled instrumentation can introduce a reaction time variable that could impose undue variability in the task results. Therefore, the creation of devices that can automatically dispense precise numbers of treats at the introduction of a predefined stimulus can be justified.

**Dispenser Operation**

The precise dispenser has a wheel on the top of it that holds up to 59 treats. The stepper motor shaft interfaces to the dispenser wheel through the center hole and by actuating the motor, the holes for the treat lines up with the treat exhaust and falls to the bowl below. When the treat goes through the exhaust it breaks an infrared break beam and the Raspberry Pi detects the treat dispensing.

**Circuit Descriptions**

**Python Class Definition**

class PreciseDispenser

**\_\_init\_\_**(loaded\_treats, timeout=1)

Initializer for the canine treat dispenser, configures control variables and Pi GPIO.

:param loaded\_treats: The number of treats that are loaded into the dispenser for this session. Defaults to max, 59 treats.

:param timeout: The amount of time before the dispenser throw an error between dispensation events

:returns: None.

**dispense\_treat**()

Function to dispense a single treat. Should not be used, use dispense\_treats for error handling.

:returns: True for successful dispensation of a single treat, False for a failure to dispense the treat.

**dispense\_treats**(num\_treats)

Dispenses the specified number of treats. If the requested number of treats is greater than the remaining treats, a ValueError is thrown. If the treat dispenser loop times out, a ValueError is thrown.

:param num\_treats: A number between 1 and the remaining number of treats.

:returns: None. Throws ValueError when an error event is encountered. **treat\_dispensed**(channel):

A callback function for when a treat is dispensed. This is detected on the falling edge of the IR break beam.

:param channel: The channel to identify the callback source.

:returns: None.

**close**()

Closes the dispenser object and cleans up the GPIO assignments.

:returns: None.

**Interfacing Example**

from precise\_dispenser\_driver import PreciseDispenser

dispenser = PreciseDispenser()

try:

dispenser.dispense\_treats(10)

print(“Successfully dispensed ten treats.”)

except ValueError as e:

print(e)

dispenser.close()

**Raspberry Pi 4 as an Experiment Platform**

The experiments on canine operant conditioning are run using a Python package called PsychoPy. It essentially acts as a development environment for psychology experiments and provides a variety of utilities for user interface development, trigger conditions, experiment logging, and so on. Our aim is to integrate this directly into a Raspberry Pi 4, to allow the entire setup to become a mobile testing unit. To do this, we will have a few requirements:

1. Wireless Access Point
   1. The Raspberry Pi 4 should host its own WiFi access point (AP), which allows the experimenter to connect to the Pi and run experiments, download results, and interact with the main system.
2. Decoupled Interfaces
   1. The screen the experimenter will use, and the screen used by the canine will be separate and require no extra hardware to achieve. By connecting to the Pi using SSH (Secure Shell), the command line can be exposed to run the experiments. The canine screen can be connected to the HDMI0 of the Pi and the SSH session can be started in a separate area.
3. Simple Setup
   1. We want to make sure this is easy to setup and maintain. This means that we would want to use robust packages such as ‘pi-ap’ and SD card backups to easily restore the device and get new devices up and running easily.

There are two ways to setup a new Raspberry Pi to operate the canine dispenser:

1. Flash a new SD card with an image from the iso\_images folder
   1. Using a Raspberry Pi with a fresh installation of Raspbian Buster, there are built-in tools to achieve [this](https://www.raspberrypi.org/documentation/linux/filesystem/backup.md). “The SD Card Copier application, which can be found on the Accessories menu of the Raspberry Pi Desktop, will copy Raspbian from one card to another. To use it, you will need a USB SD card writer. To back up your existing Raspbian installation, put a blank SD card in your USB card writer and plug it into your Pi, and then launch SD Card Copier. In the ‘Copy From Device’ box, select the internal SD Card. This could have a number of different names and may have something like (/dev/mmcblk0) in its entry, but will usually be the first item in the list. Then select the USB card writer in the ‘Copy To Device’ box (where it will probably be the only device listed). Press ‘Start’. The copy, depending on the size of the SD card, can take ten or fifteen minutes, and when complete you should have a clone of your current installation on the new SD card. You can test it by putting the newly-copied card into the Pi’s SD card slot and booting it; it should boot and look exactly the same as your original installation, with all your data and applications intact.”
2. Follow the proceeding instructions to setup the Raspberry Pi manually.
   1. Flash a new copy of Rasbian Buster onto a micro SD card and place an empty file named “ssh” onto the boot partition. There is no file extension.
   2. Connect the Raspberry Pi to your local network using an Ethernet cable and power up the Pi. You can connect a monitor, keyboard, and mouse and use ifconfig to find the IP address of the Pi.
   3. Using this IP address, go to PuTTY and type in this IP address and connect to the Pi. The default password for “pi” is “raspberry”.
   4. Once in, the [pi-ap software](https://github.com/f1linux/pi-ap) can be installed to create the access point. Note, the dog\_operant repository does contain a copy of this, but the main repository will contain the most recent version.
   5. The [dog\_operant](https://github.com/adaptive-decision-making-lab/dog_operant) repository can be cloned using the git clone command.
   6. Run the command sudo apt-get update
   7. You will need to upgrade pip using   
      python3 -m pip install --upgrade pip   
      which will allow it to index some of the following packages.
   8. Run the command   
      python3 -m pip install psychopy==3.2.4   
      which will install all the needed packages. Using the older version of PsychoPy will prevent a ‘Segmentation fault’ caused when rendering a new Window.
   9. You may get an error saying that there is no package called ‘wx’, in which case you need to run the command   
      python3 -m pip install wxPython
   10. If you want to run experiments, you will need to connect a screen to the Pi on HDMI0. An issue may present itself where no screen is presented when connected, which is caused by an incorrect /boot/config.txt file. To fix this do the following:
       1. sudo nano /boot/config.txt
       2. Add the following to the file or uncomment from the file,
          1. hdmi\_force\_hotplug=1
          2. hdmi\_drive=2
       3. If this does not fix the issue, add,
          1. hdmi\_safe=1
   11. To run an experiment, use the following command   
       export DISPLAY=:0 ; python3 dog\_operant/system\_test.py   
       Note, this works from the command line, so run this when you are connected to an SSH session and with a screen connected to HDMI0 of the Pi and the experiment will display over HDMI0. Setting the DISPLAY variable to 0 will default it to the local display.
3. To transfer the results of the experiments, copy the contents of the data directory from the dog\_operant directory. Use the following command structure:   
     
   scp -r user@ssh.example.com:/path/to/remote/source /path/to/local/destination

**Raspberry Pi Interfacing and Passwords**

The dispenser, when turned on, will create a WiFi access point called, “operant-canine-xx”, with “xx” being the number, i.e. “01”. By connecting to this, the Raspberry Pi can be accessed through SSH using a program called [PuTTY](https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html).

|  |  |
| --- | --- |
| Local Address | Password |
| pi@192.168.0.1 | cb3stevens |
| pi@adml-stevens-01 | cb3stevens |

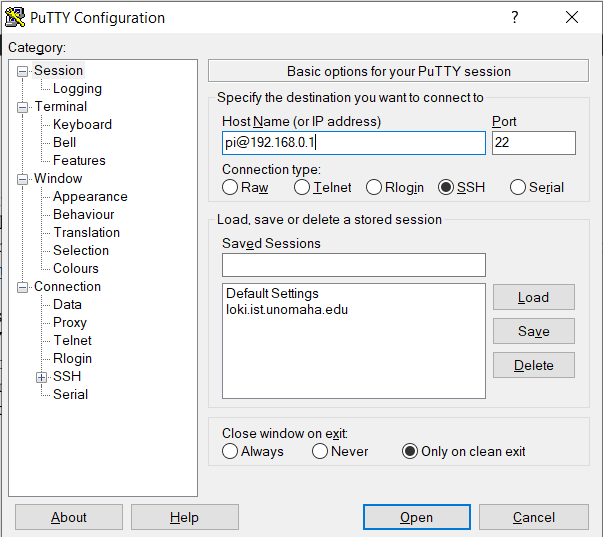


Figure 2 – PuTTY Terminal Configuration Window

**Reliability Testing**

**Bill of Materials**

**Build Instructions**

**Document Revision History**

|  |  |
| --- | --- |
| Revision, Name, and Date | Revision Description |
| V0.1, Walker Arce, 7 March 2021 | First draft creation date for the precise dispenser |
| V0.2, Walker Arce, 3 April 2021 | Added Python driver documentation |