Behav Res (2016) 48:503–509
DOI 10.3758/s13428-015-0603-2

ROBucket: A low cost operant chamber based on the Arduino microcontroller

Kavya Devarakonda[1] · **Katrina P. Nguyen**[1] · **Alexxai V. Kravitz**[1]

Published online: 28 May 2015

Abstract The operant conditioning chamber is a cornerstone of animal behavioral research. Operant boxes are used to assess learning and motivational behavior in animals, particularly for food and drug reinforcers. However, commercial operant chambers cost several thousands of dollars. We have constructed the Rodent Operant Bucket (ROBucket), an inexpensive and easily assembled open-source operant chamber based on the Arduino microcontroller platform, which can be used to train mice to respond for sucrose solution or other liquid reinforcers. The apparatus contains two nose pokes, a drinking well, and a solenoid-controlled liquid delivery system. ROBucket can run fixed ratio and progressive ratio training schedules, and can be programmed to run more complicated behavioral paradigms. Additional features such as motion sensing and video tracking can be added to the operant chamber through the array of widely available Arduino-compatible sensors. The design files and programming code are open source and available online for others to use.

Keywords Operant chamber · Operant conditioning · Open source · Arduino · Instrumental conditioning

Electronic supplementary material The online version of this article (doi:10.3758/s13428-015-0603-2) contains supplementary material, which is available to authorized users.

✉ Alexxai V. Kravitz
lex.kravitz@nih.gov

[1] Diabetes, Endocrinology, and Obesity Branch, National Institute of Diabetes and Digestive and Kidney Diseases, National Institutes of Health, Building 10-CRC, Room 5-5932, 10 Center Drive, Bethesda, MD 20892, USA

Introduction

Operant conditioning is the modification of the probability of an initially spontaneous behavior by providing reinforcing consequences contingent upon a response (Skinner, 1938). For example, a researcher may reinforce "nose-poking" behavior in mice by delivering sucrose solution each time the mouse activates a trigger with its nose. If a behavior occurs with a higher frequency following an outcome (such as delivery of sucrose) the behavior has been "reinforced," whereas if it occurs with a lower frequency it has been "punished." Operant training can be used to test learning and motivation in animals, and has been used extensively in studies of food reinforcement, drug addiction, and learning and memory. However, commercially available operant chambers cost ~ US$2000–5000 per chamber, and thus may prevent researchers and educators from performing operant behavioral experiments. Even in well-funded research environments, this cost can make high-throughput experiments involving dozens of operant chambers impractical.

To address the issue of cost, we have constructed an inexpensive and easily assembled operant chamber based on the Arduino electronics platform. Arduino input/output (I/O) boards are ideal laboratory tools because they can receive and transmit transistor-transistor-logic (TTL) signals on a millisecond timescale without a computer interface (D'Ausilio, 2012). Arduino programs – called "sketches" – are written in a simple language with the Arduino integrated development environment (IDE) software. The bootloader on the Arduino board allows users to upload sketches from a computer via a universal serial bus (USB). In addition, there are many publicly available, open-source libraries, which are sets of programs that define common functions. Libraries allow users to more easily execute complex commands thus expanding the capabilities of the Arduino programming language and hardware.

The Arduino platform also has an array of widely available sensors and "shields" that have been developed by the Arduino community. Shields are modular circuit boards that plug into the base Arduino board to extend its functionality. With these, users can custom-design hardware to suit their needs. This can include features that many commercial operant chambers do not offer, such as touchscreens, motion detectors, and Bluetooth or Wi-Fi connectivity.

The operant chamber we designed, the Rodent Operant Bucket (ROBucket), combines an Arduino Uno with three photo interrupter sensors, a secure digital (SD) card shield, and a liquid crystal display (LCD) keypad shield. ROBucket costs US$150 and takes approximately 90 min to build. Required parts and circuitry can be obtained "off the shelf" (Table 1), with the exception of one three-dimensional (3D) printed housing for the nose pokes and drinking well, which is not included in our price estimate. Design files can be found on the authors' website (http://www.niddk.nih.gov/research-funding/at-niddk/labs-branches/diabetes-endocrinology-and-obesity-branch/eating-addiction-section/rodent-operant-bucket/Pages/default.aspx). The programming code we provide can be used to train outcome retrieval (magazine training), and to train mice to nose-poke for sucrose solution on a fixed ratio 1 (FR1), FR5, or progressive ratio (PR) schedule of reinforcement (Supplementary Materials).

Table 1 List of "off the shelf" parts used to build the ROBucket

Part	Quantity	Total Price (US$)
Four-quart square container (6.5" × 6.5" × 8")	1	$5.97
Lid for 2- and 4-quart containers	1	$1.42
Arduino UNO - R3	1	$24.95
LCD keypad shield for Arduino	1	$14.55
Adafruit assembled data logging shield for Arduino	1	$19.95
Kingston 4GB microSD high capacity (microSDHC) card	1	$8.99
Photo interrupter board with GP1A57HRJ00f	3	$14.97
Sainsmart 2-channel 5V relay module	1	$18.99
12 VDC solenoid valve	1	$3.25
Clear acrylic squares ½ " acrylic square	1	$0.24
JST SH jumper 3-wire assembly – 8"	3	$4.05
Faceplate, 3-port	1	$7.91
Machine screw,flat,4-40 × ½ L, nylon	8	$0.32
Hex nut, machine screw, 4-40, black, nylon	12	$0.32
Wall adapter power supply – 9VDC 650 mA	1	$5.95
DC barrel jack adapter – female	1	$2.95
½" straight cannula blunt end tip (15 gauge)	1	$0.24
Male luer with lock ring × 1/8" hose barb, nylon	1	$0.34
Male luer with lock ring × female luer coupler, PC	1	$0.56
BD 20mL syringe with Luer-Lok tip	1	$0.51
Masterflex platinum-cured silicone tubing, L/S 16, 25 ft.	0.5 Ft	$1.18
Stor-All broom clip	2	$3.38
Jumper wires premium 6" F/F	12	$2.99
Wing nut,6-32	2	$1.07
Machine screw, flat, SS, 6-32 × 1.5 L	2	$0.29
2-way 2.1 mm DC barrel jack splitter squid	1	$2.95
0.1" (2.54 mm) crimp connector housing: 1 × 1-pin	9	$0.21
Female crimp pins for 0.1" housings	9	$0.54
3D printed housing	1	variable
Total		$149.04

Overview of the device

The ROBucket chamber consists of a four-quart square plastic container (6.5 in × 6.5 in × 8 in.) and a simple input/output circuit (Fig. 1). The chamber is equipped with three photo interrupter sensors, which detect interactions with the nose pokes and drinking well. The nose pokes flank the drinking well and are indistinguishable from each other except that responding on the designated "active" poke results in sucrose delivery whereas responding on the "inactive" poke produces no consequence. Responses on both pokes and the drinking well are logged to the SD card. The drinking well is distinctive in that a blunt needle descends into the opening, which is enclosed from the bottom with a clear plastic square.

The photo interrupter sensors send a constitutively "high" (5V) TTL signal to the Arduino. A nose-poke response involves a mouse breaking the infrared photobeam of a photo interrupter board with its nose. When a mouse responds on a nose poke, a "low" signal is sent to the Arduino, triggering the sucrose delivery process. To avoid false responses, all reinforcement schedules require the mouse to remove his nose between pokes, and the FR1 program includes a 1-s timeout in which the nose poke is inactive following each active signal.

After receiving a "low" signal from the active nose poke, Arduino sends a 5V output signal to trigger a solenoid valve through a relay module. The signal causes the solenoid to open for 75 ms, allowing approximately 60 μL of sucrose solution to be dispensed into the drinking well. As each behavioral event occurs, the Arduino timestamps the event to a file on the SD card and updates the trial information displayed on the LCD keypad shield.

Hardware

There are six key pieces of hardware in ROBucket (Fig. 2a). The Arduino Uno R3 board is controlled by the ATmega328 microcontroller chip, which has 31.5 kilobytes (KB) of available memory. It has 14 digital I/O pins and six analog input

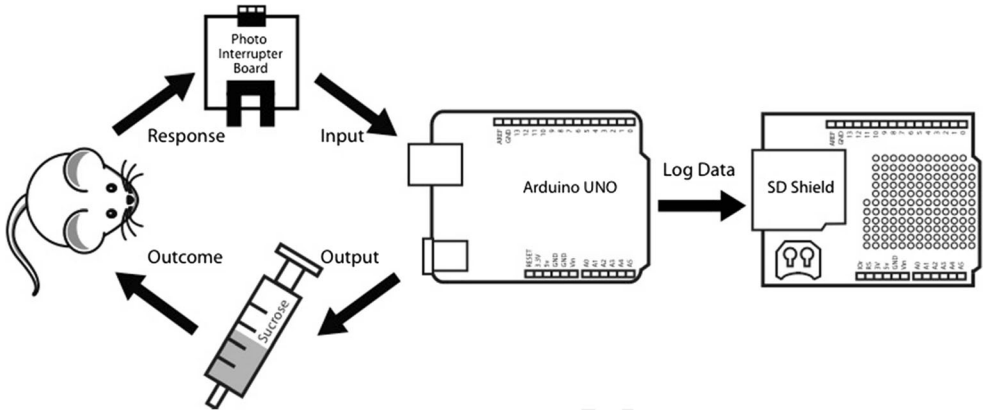

Fig. 1 Diagram of the ROBucket's electronic circuit. A photo interrupter sensor detects a mouse's response (i.e., nose poke) and sends a signal to a digital pin on the Arduino Uno. Based on the chosen training schedule, the Arduino Uno sends a signal to the relay and solenoid valve, resulting in delivery of sucrose. As each behavioral event occurs, the Arduino logs this information via a data logging module (secure digital (SD) shield)

pins. The Arduino Uno board can respond to a single input in less than 1 ms and to multiple conditions in 100–200 ms, depending on the task and programming code (D'Ausilio, 2012). The Adafruit data logging shield works with FAT16 or FAT32 formatted SD cards, and includes a real-time clock (RTC) to allow accurate date and time stamping of events. The Arduino communicates with the SD card interface of the data logging shield through a serial peripheral interface (SPI), which requires the master output slave input (MOSI); master input slave output (MISO); serial clock (SCK); and chip select (CS) pins (these are digital pins 11, 12, 13, and 10 on the Arduino Uno, respectively). The RTC portion of the data logging shield retrieves and sets time through the I^2C bus of the Arduino board. The I^2C bus allows multiple devices to communicate with the microcontroller through the same two lines of communication, a clock signal (SCL) and a data line (SDA). The LCD keypad shield has a 16 × 2 character backlight LCD and five keys (Select, Up, Right, Down, and Left). The keypad interface uses analog pin 1 and the LCD screen uses digital pins 4–9 on the Arduino Uno. Digital pin 10 controls the backlight of the screen and can be shared with the data logging shield.

The three photo interrupter boards act as the "active" and "inactive" nose-poke sensors and a drinking well entry sensor to provide input to the Arduino Uno. The photo interrupter boards have an infrared LED sensor to detect when an object enters the 10 mm gap in the arms at a temporal resolution of 3 μs. The Arduino's 5V output signal is detected by the relay module, which controls the opening of a 12V DC solenoid valve. When the solenoid valve is open, sucrose solution is able to flow through the valve into the drinking well inside the chamber. Off-the-shelf parts are used to hold the main hardware in place and facilitate their functions. For step-by-step directions on how to construct a ROBucket, please visit the authors' website (http://www.niddk.nih.gov/research-funding/at-niddk/labs-branches/diabetes-endocrinology-and-obesity-branch/eating-addiction-section/rodent-operant-bucket/Documents/ROBucket%20construction%20instructions.pdf).

Figure 2b shows the wiring of the ROBucket hardware and Fig. 2c shows a fully constructed ROBucket. In addition to the main hardware, a 3D printed plastic housing is combined with a stainless steel three-port Ethernet faceplate to enclose the nose pokes and drinking well inside the bucket (Figs. 2d–e). The part was designed using the free online software, Tinkercad (Autodesk, Inc., San Rafael, CA, USA), and printed using the Makerbot Replicator 2X Desktop 3D Printer (MakerBot Industries, Brooklyn, NY, USA). The STL file is available on the authors' website, which also includes a link to the CAD model.

Software

The ROBucket sketch requires the Arduino LiquidCrystal library to control the LCD keypad shield, the Arduino SD library to write to the data logging shield, and the Arduino Wire library to use the I^2C bus; all three libraries are already installed on the Arduino IDE, available for download at http://arduino.cc/en/Main/Software. The ROBucket sketch also requires one third-party library, *RTClib* (available for download at the Adafruit Industries website and on the authors' website), to communicate with the RTC on the data logging shield.

Once the ROBucket sketch has been uploaded to the Arduino Uno and an SD card has been inserted into the data logging shield, the LCD screen displays a menu of four reinforcement schedules that can be selected by pressing the corresponding keypad button. The four programs include: (1) magazine training, which delivers sucrose solution on a variable time interval between 45 and 75 s; (2) the FR1 program, which delivers sucrose solution when a mouse pokes the active poke, with a timeout of 1 s between each response; (3) the FR5 program, which requires the mouse to poke five times for sucrose delivery; and (4) the PR program, which progressively

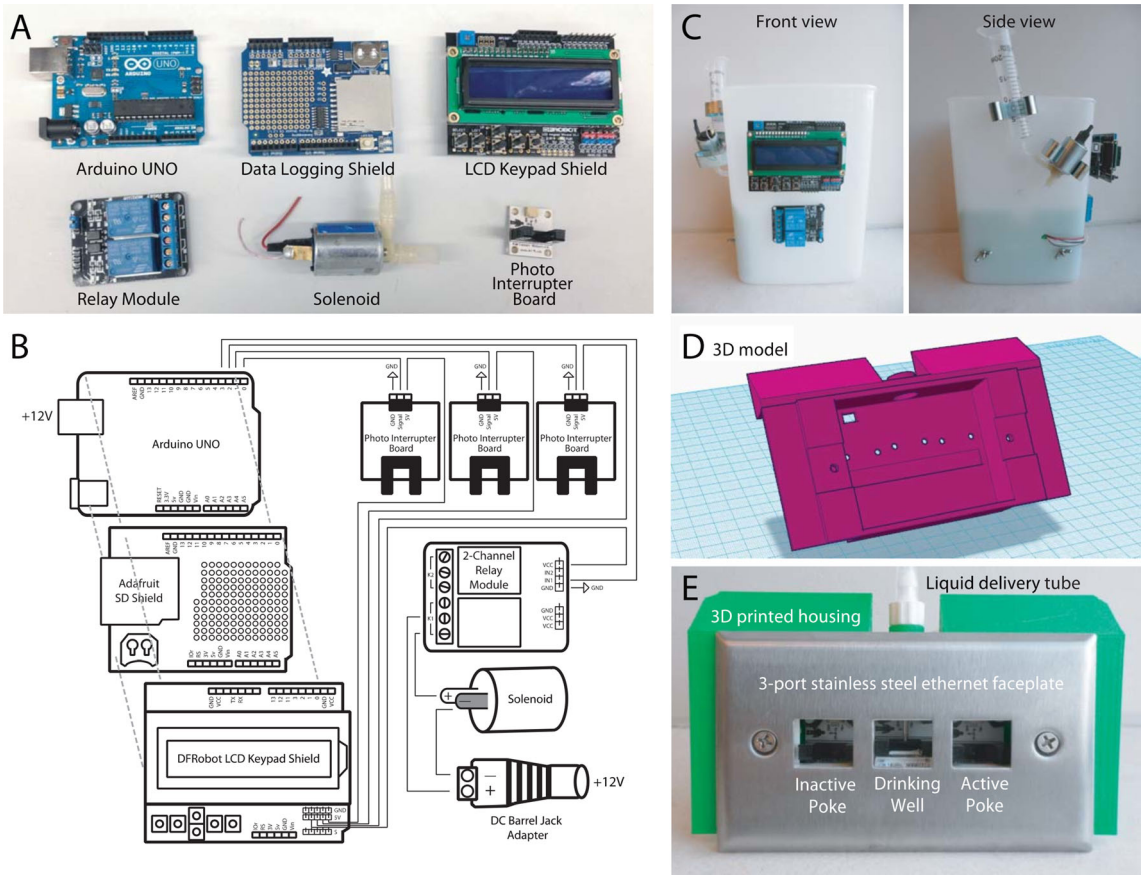

Fig. 2 ROBucket hardware. (**a**) The electronic components of ROBucket (clockwise from top left): Arduino Uno R3 (Smart Projects, Ivrea, Italy), data logging shield (Adafruit Industries, New York, NY, USA), LCD keypad shield (DFR Robotics Ltd., Shanghai, China), photo interrupter board (Karlsson Robotics, Jupiter, FL, USA), 12VDC solenoid (American Science & Surplus, Skokie, IL, USA), and two-channel 5V relay module (Sainsmart, Leawood, KS, USA). (**b**) The wiring schematic for the electronic components of ROBucket. (**c**) Prototype of the assembled ROBucket, which took 90 min to assemble and costs US$149. (**d**) The 3D model of the custom-designed nose poke housing and (**e**) a 3D printed plastic housing combined with a stainless steel three-port Ethernet faceplate to enclose two nose pokes and drinking well

increases the number of pokes required for sucrose delivery following the equation, $r = 5e^{0.2n} - 5$ (rounded to the nearest integer; Richardson and Roberts, 1996). The PR program ends if the mouse does not poke for 10 min, whereas the other three programs continue until stopped by the user. Pressing the "Select" button on the LCD keypad shield stops the current trial and ends data logging. The "Reset" button begins a new trial and resets all counts and timers except for the RTC.

During a trial, ROBucket's LCD screen displays the current program, the trial duration, the number of sucrose deliveries, the number of active and inactive pokes, and the number of times the mouse has entered the drinking well. Additionally, ROBucket saves this information to a comma-separated values (CSV) file on the SD card in the data logging shield. When sucrose is delivered, or when the mouse nose-pokes or enters the drinking well, the Arduino writes a new line to the CSV file with the updated counts, the date and time, and the number of pokes that were necessary to earn each sucrose delivery. At the end of a trial, the final counts and the duration of the trial are written to the CSV file. If a new trial is started at this point, a new row of column titles are written to the CSV file, indicating the start of a new trial. The data acquisition format was designed to facilitate graph-making and analysis in common statistical software packages (e.g., Microsoft Excel).

Behavioral experiments

Methods

To evaluate how ROBucket can be used to train mice to respond for a sucrose solution, eight adult C57BL/6 male mice were trained in a ROBucket that had an active and an inactive poke. Nose-poking in the active poke resulted in the delivery of a 20 % sucrose solution, whereas responding on the inactive poke had no consequence. The training sessions were 1 h

in duration and occurred 1–2 h before the onset of the animals' dark cycle. Prior to the start of training, the mice were food restricted to maintain 85–95 % of free-feeding body weight. Mice were trained on the FR1 program until they met the acquisition criteria by earning more than 20 sucrose deliveries in a session and exhibiting discrimination greater than 3:1 for the active versus inactive nose pokes (Sharma et al., 2012). Once the mice met the acquisition criteria for three consecutive days, four of the mice graduated to the FR5 reinforcement schedule for 3 days, followed by the PR program. On the PR schedule, the number of sucrose deliveries earned during a session was called the breakpoint. The breakpoint is used to measure the strength of an animal's motivation for an outcome because it represents how hard the animal is willing to work for the outcome. The mice were trained on the PR program until their breakpoints were stable (±1) for 3 consecutive days.

To verify that ROBucket can assess changes in motivation for sucrose, we performed a satiety test, which decreases both "liking" and "wanting" for sucrose (Havermans et al., 2009). Briefly, the mice were allowed *ad libitum* access to sucrose solution in a novel home cage for 1 h prior to being placed in a ROBucket and reinforced on a PR schedule. The PR breakpoint was calculated as before. Data were analyzed by repeated measure two-way ANOVA and two-tailed paired t-tests (statistiXL, Nedlands, WA, Australia).

Results

Six of the eight mice achieved acquisition criteria after eight training sessions (Fig. 3). Two mice did not achieve acquisition criteria by day 9 and were excluded from the study (#637 and #646). Mice required between 5 and 8 days, with an average of 6.3 days, to meet the acquisition criteria. These results are consistent with operant training on commercially sold systems, and the acquisition curves of the reinforced mice were similar to those of mice trained in commercially sold chambers (Haluk and Wickman, 2010; Sharma et al., 2012).

The six mice that achieved acquisition criteria increased responding on the active poke from an average of 16 ± 4.0 pokes on day 1 to 43 ± 7.5 responses by day 5 of training (Fig. 4a; data reported as mean ± standard error of the mean). The mice maintained a low basal level of responding on the inactive poke (9 ± 3.1 responses on day 1 vs. 13 ± 6.2 on day 5). There were significant main effects of session ($F = 9.708$, p

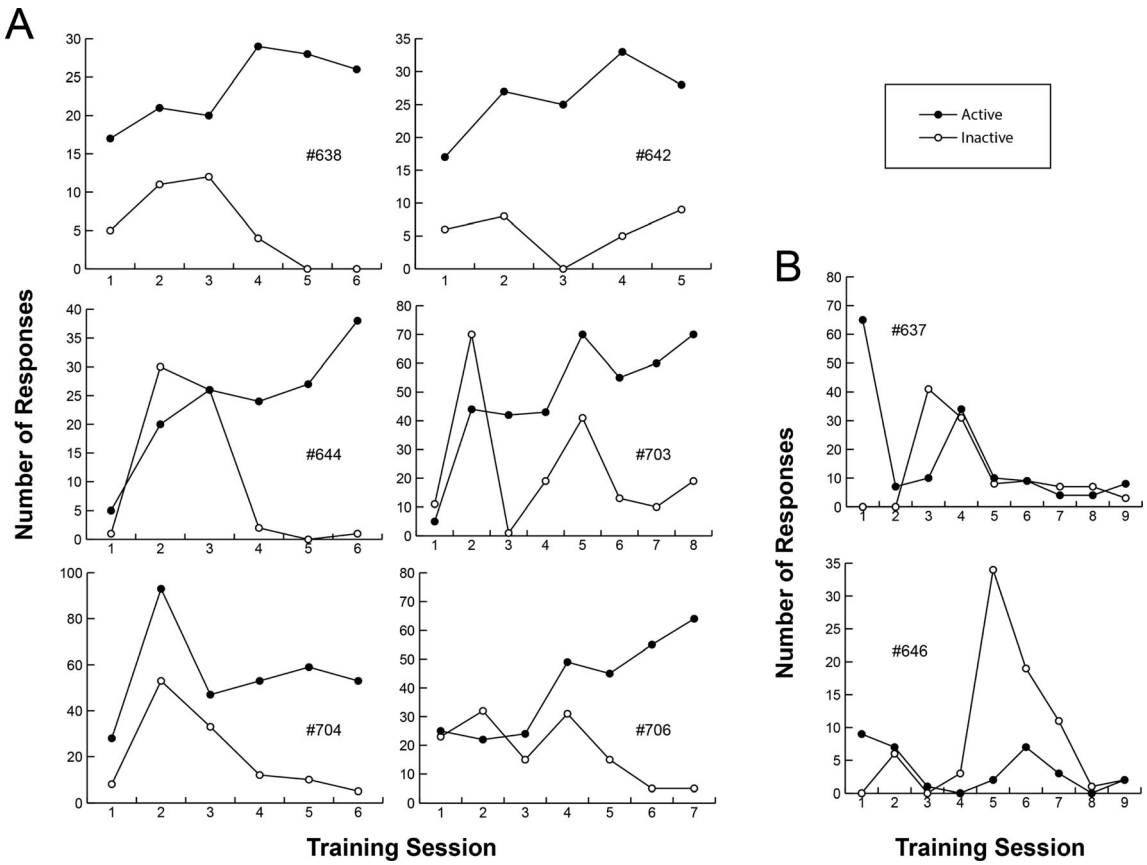

Fig. 3 Six out of eight mice met the acquisition criteria within 8 days. Number of responses on the active (closed circles) and inactive (open circles) during each 1-h training session for the mice that (**a**) learned the operant task and (**b**) did not learn the task after 9 days. Each graph is labeled with the animal's identity

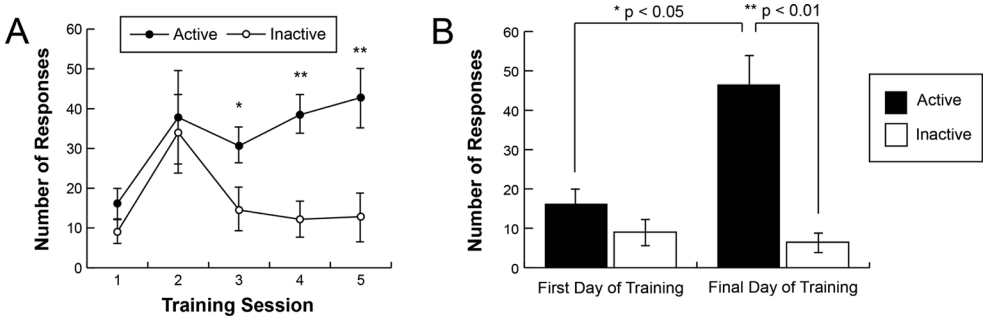

Fig. 4 The ROBucket can train mice to respond for sucrose. (**a**) Number of responses on the active (closed circles) and inactive (open circles) during the first five 1-h training sessions. (**b**) Number of responses on the active (black bars) and inactive (white bars) during the first and last day of each animal's training. Data are expressed as mean ± SEM. Asterisks indicate significant within-subject difference (* $p < 0.05$, ** $p < 0.01$), analyzed by a *post hoc* two-tailed paired t-test

= 0.006) and poke ($F = 6.149$, $p = 0.033$), and a significant interaction effect ($F = 5.902$, $p = 0.021$), with significant within-subject differences at days 3, 4, and 5 (Fig. 4a). Comparing each animal's first and final training sessions (Fig. 4b), there were significant main effects of session ($F = 9.169$, $p = 0.013$) and poke ($F = 22.756$, $p = 0.001$), and a significant interaction effect ($F = 12.759$, $p = 0.005$; Fig. 4b). The mice that met the acquisition criteria significantly increased responding on the active poke between the first and last day of training, and on the final day of training there was a significant difference between responding on the active versus the inactive nose poke.

On the PR reinforcement schedule, the mice had an average breakpoint of 7.3 ± 1.2 sucrose deliveries, which required at least 49 responses in total on the active nose poke (Fig. 5). After an hour of *ad libitum* access to the sucrose solution (satiety test), the mice had significantly lower breakpoints, with an average of 3.5 ± 0.9 (paired t-test, $p = 0.021$). These data demonstrate that the mice's responses were due to their motivation to obtain sucrose because devaluation of the sucrose resulted in decreased responding on the PR task.

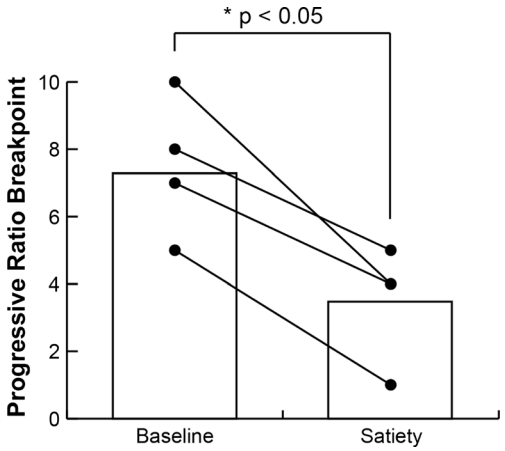

Fig. 5 Devaluation of sucrose decreases responding on a progressive ratio reinforcement schedule. Progressive ratio breakpoint at baseline and after 1-h *ad libitum* access to 20 % sucrose solution in the home cages. Bars represent mean breakpoint values, whereas closed circles represent individual subject data. An asterisk indicates significant within-subject difference (* $p < 0.05$), analyzed by a two-tailed paired t-test

Discussion

ROBucket is an inexpensive and easily assembled open-source operant chamber based on the Arduino microcontroller platform. The primary advantage of ROBucket over commercially available operant training systems is the low cost. Each ROBucket costs less than US$150, which is 10–20 times cheaper than commercially available operant chambers. Although other homemade, open-source operant chambers have been developed, ROBucket is unique because of its simplicity and versatility (Pineño, 2014). ROBucket was designed to be easily produced by research laboratories and classrooms using almost entirely "off the shelf" parts. The parts list, design files, and detailed protocols for constructing ROBucket are available on the authors' website. For those familiar with electronics and tools, ROBucket is easily assembled in 90 min, and making multiple chambers simultaneously reduces the per unit construction time.

Furthermore, the ROBucket sketch we provide is easily modified to suit specific experiments or laboratories. Additional features can be added to ROBucket through the array of widely available Arduino-compatible accessories. For example, a motion sensor could provide a measure of the mouse's total activity during operant training sessions or a TTL-triggered camera could record video while a trial is active. Lastly, other models of the Arduino microcontroller provide more I/O pins and memory (up to 54 pins and 248 KB on the Arduino Mega 2560), allowing complex arrays of sensors and other hardware to be controlled by a single Arduino. Specifically, ROBucket can be redesigned to provide two different liquid outcomes with the addition of another set of nose pokes, a drink well, and delivery system. Or ROBucket can be used for Pavlovian conditioning with the addition of a tone generator or light cue. Because Arduino is an open-source platform, the additional functionality

suggested above would not significantly add to the cost of ROBucket.

However, there are some limitations with a do-it-yourself device like ROBucket. The most prominent is the labor costs. Whereas commercially sold operant chambers need a large initial investment of time with little additional effort to set up new experiments, each ROBucket chamber requires construction time and some knowledge of electronics. Making a ROBucket requires some specialized equipment, such as soldering tools, which may not be found in all laboratories. Additionally, although the ROBucket sketch can be modified, commercial operant chambers may be easier to program with complex training paradigms.

Another limitation of ROBucket is the reliability of the electronic parts and the variability between buckets. Because the parts used to build ROBucket are inexpensive, certain components can have inconsistent performance. For example, although the ROBucket code specifies that the solenoid should open for 75 ms, solenoids may vary in the amount of sucrose they dispense during that time. In our experience, the amount of sucrose delivered by the solenoids can vary by 20–30 μL, and the amount delivered by the same solenoid may vary by 10 μL. To reduce variability between buckets, users can calibrate each solenoid delivery system by changing the amount of time each solenoid is open. This calibration only requires changing the value of a single variable in the programming code. We recommend testing each ROBucket after it is made and periodically during training to avoid issues with reliability. If a particular experiment requires greater precision, users can purchase more expensive solenoids, which will have more consistent performance.

In summary, we believe there is a role for inexpensive, user-constructed ROBuckets in research projects that require a low price per chamber, or in educational settings in which the construction itself may be an asset for introducing students to electronics as well as animal behavior training.

Acknowledgments This work was supported by the National Institute of Diabetes and Digestive and Kidney Diseases Intramural Research Program. Thanks to Daniel Serven and Verma Walker for their assistance with designing and printing the 3D printed housing, the members of the Kravitz laboratory for input on the design and use of the ROBuckets, and Michael J. Krashes for inspirational discussions. 3D printing services were provided by the National Institutes of Health Technology Sandbox and the National Institutes of Health Library.

References

D'Ausilio, A. (2012). Arduino: A low-cost multipurpose lab equipment. *Behavior Research Methods, 44,* 305–313. doi:10.3758/s13428-011-0163-z

Haluk, D. M., & Wickman, K. (2010). Evaluation of study design variables and their impact on food-maintained operant responding in mice. *Behavioural Brain Research, 207,* 392–401. doi:10.1016/j.bbr.2009.10.025

Havermans, R. C., Janssen, T., Giesen, J. C. A. H., Roefs, A., & Jansen, A. (2009). Food liking, food wanting, and sensory-specific satiety. *Appetite, 52,* 222–225. doi:10.1016/j.appet.2008.09.020

Pineño, O. (2014). ArduiPod Box: A low-cost and open-source Skinner box using an iPod Touch and an Arduino microcontroller. *Behavior Research Methods, 44,* 196–205. doi:10.3758/s13428-013-0367-5

Richardson, N. R., & Roberts, D. C. (1996). Progressive ratio schedules in drug self-administration studies in rats: A method to evaluate reinforcing efficacy. *Journal of Neuroscience Methods, 66,* 1–11. doi:10.1016/0165-0270(95)00153-0

Sharma, S., Hryhorczuk, C., & Fulton, S. (2012). Progressive-ratio responding for palatable high-sugar food in mice. *Journal of Visualized Experiments, 63*, e3754. doi:10.3791/3754

Skinner, B. F. (1938). *The behavior of organisms: An experimental analysis*. New York: Appleton-Century.