Revista Mexicana de Análisis de la Conducta • Mexican Journal of Behavior Analysis
2017 | Núm. 2 • septiembre | No. 2 • September | Vol. 43, 242 - 253 |
https://doi.org/10.5514/rmac.v43.i2.62315

NEW TECHNOLOGIES APPLIED TO OPERANT RESEARCH: INEXPENSIVE PHOTOCELLS FOR THE ARDUINO-VISUAL BASIC INTERFACE

NUEVAS TECNOLOGÍAS APLICADAS A LA INVESTIGACIÓN OPERANTE: FOTOCELDAS DE BAJO COSTO CON LA INTERFAZ ARDUINO-VISUAL BASIC

Rogelio Escobar and Nadia Santillán
National Autonomous University of Mexico

Abstract

In recent years, technological advances in electronics have been applied for designing flexible and inexpensive interfaces for recording and controlling events in operant research. One example, tested for accuracy and reliability, uses the popular Arduino microcontroller boards in combination with Visual Basic Express Edition programming. This paper describes one of the applications of the Arduino-Visual Basic interface to record movement using photocells. Each photocell uses a phototransistor, and an infrared LED, and allows recording the interruption of the light emitted by the infrared LED. Such interruptions are detected as responses, and are used to schedule experimental events in Visual Basic programs. These photocells can be used in diverse experimental settings under varied conditions of illumination

Rogelio Escobar and Nadia Santillán, Laboratorio de Análisis de la Conducta, Facultad de Psicología, Universidad Nacional Autónoma de México.

This project was supported by grant IT300517 awarded by PAPIIT, DGAPA UNAM to the first author. Those interested in using the photocell described can contact the first author for technical support (rescobar@unam.mx). The authors thank Alexis Pérez for his assistance with previous versions of the programs.

and distances without changing the hardware setup. Programs for calibrating and using the photocells can be freely downloaded and modified by the users. This set-up can also be used for recording discrete responses using other sensors like force and flexion sensors.

Keywords: Arduino, Visual Basic, interface, instrumentation, photocells, analog sensors

Resumen

En años recientes los avances en electrónica se han aplicado en el diseño de interfaces flexibles y de bajo costo para registrar y controlar eventos en investigación operante. Un ejemplo, probado en términos de precisión confiabilidad, usa las populares tarjetas de microcontrolador Arduino en combinación con programación en Visual Basic Express Edition. Este trabajo describe una de las aplicaciones de la interfaz Arduino-Visual Basic para registrar movimientos usando fotoceldas. Cada fotocelda usa un fototransistor y un LED infrarrojo y permite registrar la interrupción de la luz emitida por el LED infrarrojo. Tales interrupciones se detectan como respuestas y se usan para generar eventos experimentales en los programas en Visual Basic. Estas fotoceldas pueden usarse en diversos ambientes experimentales y bajo diferentes condiciones de iluminación y distancias sin tener que hacer modificaciones en el equipo. Los programas para calibrar y usar las fotoceldas pueden descargarse de manera gratuita y pueden ser modificados por los usuarios. Este diseño puede usarse también para registrar respuestas discretas con otros sensores como los sensores de flexión y de fuerza.

Palabras clave: Arduino, Visual Basic, interfaz, instrumentación, fotoceldas, sensores analógicos

Instrumentation in behavior analysis involves the design and use of instruments for studying interactions between behavior and environment. Because of the experimental nature of behavior analysis, demonstration of repeatable phenomena is essential for the advancement of the discipline. While many behavioral phenomena can be observed without the aid of instruments, using instruments has been advantageous in behavior analysis, as in other scientific endeavors, for three main reasons: (1) instruments expose environment – behavior relations that are not easily, or cannot be observed at all, with the naked eye, (2) they reduce human error in recording and presenting events, and (3) they allow for automation. In behavior analysis, the use of instruments has evolved from the early use of salvaged relays and motors to the development of standard commercial equipment that uses computers and high-resolution interfaces, and can be programmed with a custom-made programing language (see Escobar, 2014 for a review; Manabe, 2017, this issue).

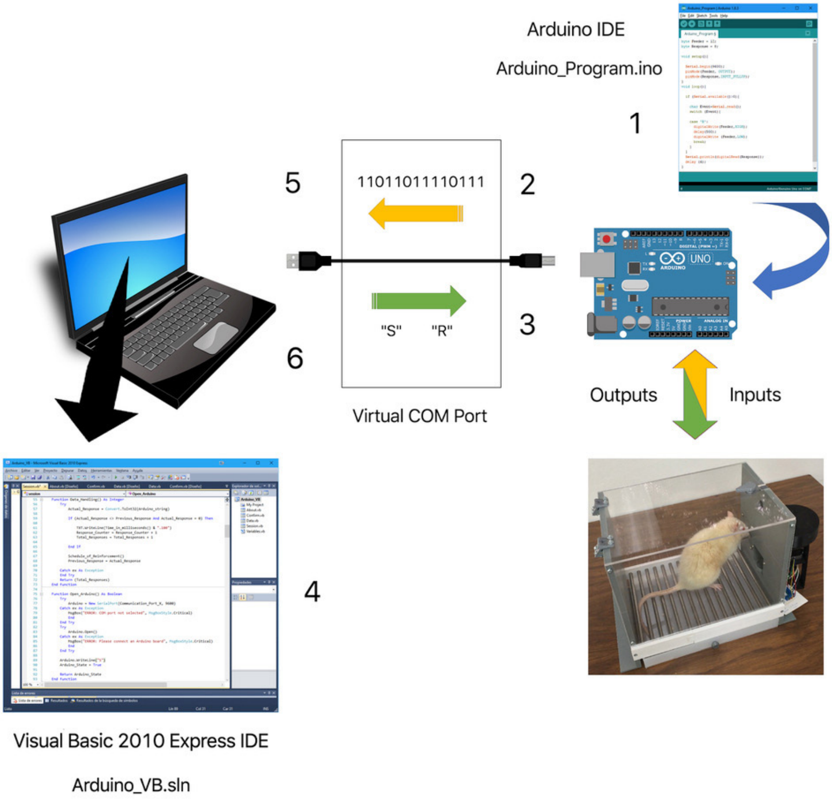
Although commercial equipment simplifies enormously the tasks of experimenters it has been associated with high prices and lack of flexibility. In developing countries, like Mexico, shipping costs and taxes added to the price of the equipment have resulted in limited access to such equipment. While many laboratories in the main universities in the country like the National Autonomous University of Mexico, University of Guadalajara, and University of Veracruz, thanks mainly to research grants provided by the National Council of Science and Technology (CONACYT), are equipped with commercial equipment, acquiring this equipment without funding is almost impossible. Therefore, young researchers, particularly in small Universities, do not have, or must wait years, before having access to such equipment. Another problem associated with commercial equipment, especially in developing countries, is lack of flexibility. Once the equipment is acquired, modifying it or acquiring additional components is intricate. Grants provide resources to acquire equipment but additional components, sometimes required after grants end, are difficult to obtain. Although the equipment could be modified, such practice is discouraged by the risk of making the equipment unusable. Thus, it is preferable to adjust research questions to fit the equipment at hand, over modifying such valuable devices.

The problems associated with commercial equipment used for operant research has been the impetus for designing inexpensive, easy-to-use equipment using recent advances in electronics (e.g., Escobar & Pérez-Herrera, 2015; Varnon & Abrahamson, 2013). Escobar and Pérez-Herrera (2015), for example, described an interface for controlling operant-conditioning chambers using the ubiquitous Arduino microcontroller boards and Visual Basic (VB) Express Edition programming. Microcontroller boards can be compared to an inexpensive miniature computer; they are programmed to generate outputs and read input values. These boards can be easily connected to a variety of output devices like LEDs and motors, and to a variety of input devices like switches. Arduino UNO boards are the most popular microcontroller boards.

An Arduino microcontroller board is loaded with one relatively small program (32 kB maximum in the Arduino UNO board) written in C-like programming language. The program executes repeatedly whenever the board is powered on, and is designed for a specific input/output setting known as embedded system. An application of microcontroller boards are, for example, alarm systems, in which a tone is generated if movement is detected. In this case, microcontroller boards generally are battery powered and are connected to small screens to control the state of the alarm. Although microcontroller boards are generally used in independent battery-powered systems like the one described, they can also be used in combination with computers. Arduino boards can be connected to a computer using a USB port. The USB port powers the Arduino board and allows bidirectional communication with the board using a Virtual Serial Port or Virtual COM port (VCP). A VCP can be thought of as a space in which devices can write and read data. Each Arduino board is assigned automatically a VCP (e.g., COM3) when it is connected to a computer. This VCP is always the same for the same Arduino board in the same computer. If a computer program, written in any language, sends data to a VCP, it can be read by the Arduino board. Also, if the Arduino board sends data to the VCP it can be read using a program accessing the VCP.

Escobar and Pérez-Herrera (2015) designed an interface for controlling operant-conditioning chambers that combined Arduino boards with Visual Basic Express Edition programming. Figure 1 shows schematically how the Arduino-Visual Basic interface works. An Arduino program is loaded to the Arduino board (1). The program sends continuous signals to the assigned VCP through the USB port representing the state of a digital input (e.g., a lever) (2). The Arduino program also reads data from the VCP and transform it into outputs in the experimental chamber (3). The schedule of reinforcement, written in Visual Basic (4), counts responses by reading the state of the inputs in the same VCP assigned to the Arduino board (5), and signal outputs by sending data (e.g., character "R") to this VCP (6). Therefore, two programs are required: one C-like program running in the Arduino board, and a Visual Basic program executing the schedule of reinforcement running on a PC. Some advantages of the interface described by Escobar and Pérez-Herrera over other low-cost alternatives to commercial equipment are that Arduino boards are accurate and easy to find worldwide, and Visual Basic programming has been used previously for generating a variety of operant procedures (e.g., Dixon & McLin, 2003). A limitation, however. is that it only works on PCs running Windows Vista or above.

Escobar and Pérez-Herrera (2015) provided links to download free Arduino and Visual Basic programs. The programs generate four basic schedules of reinforcement or concurrent schedules for two responses using digital input devices like levers or keys. These devices are known as digital because their state can be only


Arduino IDE
Arduino_Program.ino
1
2
3
4
5
6
11011011110111
"S"
"R"
Virtual COM Port
Outputs
Inputs
UNO
ARDUINO
Visual Basic 2010 Express IDE
Arduino_VB.sln
Arduino_Program | Arduino 1.8.3
File Edit Sketch Tools Help
Arduino_Program §
Arduino_VB - Microsoft Visual Basic 2010 Express
Archivo Editar Ver Proyecto Depurar Datos Herramientas Ventana Ayuda
Session.vb* About.vb [Diseño] Confirm.vb Data.vb [Diseño] Data.vb Confirm.vb [Diseño]
Explorador de sol...
Arduino_VB
My Project
About.vb
Confirm.vb
Data.vb
Session.vb
Variables.vb
Propiedades
Lista de errores
Lista de errores Resultados Resultados de la búsqueda de símbolos

```
byte Feeder = 12;
byte Response = 8;

void setup(){

  Serial.begin(9600);
  pinMode(Feeder, OUTPUT);
  pinMode(Response, INPUT_PULLUP);
}
void loop(){

  if (Serial.available()>0){

    char Event=Serial.read();
    switch (Event){

    case 'S':
      digitalWrite(Feeder,HIGH);
      delay(500);
      digitalWrite (Feeder,LOW);
      break;
    }
  }
  Serial.println(digitalRead(Response));
  delay (6);
}
```

```
    Function Data_Handling() As Integer
        Try
            Actual_Response = Convert.ToInt32(Arduino_string)

            If (Actual_Response <> Previous_Response And Actual_Response = 0) Then

                TXT.WriteLine(Time_in_milliseconds() & ".100")
                Response_Counter = Response_Counter + 1
                Total_Responses = Total_Responses + 1

            End If

            Schedule_of_Reinforcement()
            Previous_Response = Actual_Response

        Catch ex As Exception
        End Try
        Return (Total_Responses)
    End Function

    Function Open_Arduino() As Boolean
        Try
            Arduino = New SerialPort(Communication_Port_X, 9600)
        Catch ex As Exception
            MsgBox("ERROR: COM port not selected", MsgBoxStyle.Critical)
            End
        End Try
        Try
            Arduino.Open()
        Catch ex As Exception
            MsgBox("ERROR: Please connect an Arduino board", MsgBoxStyle.Critical)
            End
        End Try

        Arduino.WriteLine("S")
        Arduino_State = True

        Return Arduino_State
    End Function
```

Figure 1. Arduino-Visual Basic interface described by Escobar & Pérez-Herrera (2015). An Arduino program and a Visual Basic program communicate using a Virtual COM Port. An Arduino UNO board is connected to an operant-conditioning chamber.

on or off. As noted by Escobar and Pérez-Herrera, a variety of stimuli-producing devices can be added simply by connecting the device and declaring the output in the Arduino and Visual Basic programs. Also, a variety of sensors can be used with minor modification.

This paper describes one such example showing how inexpensive photocells, also known as infrared (IR) sensors, can be assembled in a few minutes, and used to record responses with the Arduino-Visual Basic interface. Although many tutorials are available on websites like YouTube and Instructables showing how photocells can be connected to an Arduino board, none of those are specific for recording behavior and, furthermore, do not show how these photocells can be used in conjunction with schedules of reinforcement. This paper describes how this can be done and provides links to download free programs to do it.

Photocell Use in Behavioral Research

One common arrangement of photocells consists of aligning an IR light emitting diode (LED) and a light-receiving device (e.g., a phototransistor), such that when the beam of light from the LED is interrupted, the phototransistor changes the current received by an input in the control equipment and a response is recorded. Photocells are helpful for recording responses in a variety of species and scenarios. For example, photocells have been used for recording nose poking in rats (e.g., Antonitis, 1951), swimming through a ring in *Betta splendens* fish (e.g., Lattal & Metzger, 1994), attack in pigeons (e.g, Muller & Cheney, 1975), and motor activity in rats (e.g., Ericson, Samuelsson, & Ahlenius, 1991).

Low-cost photocells have been described previously. For example, Batson and Turner (1986) described a photocell with an IR LED and a phototransistor. Although this photocell was inexpensive and easy to assemble, it required additional circuits and hardware to be connected to an Apple II computer. Similarly, Clarke, Smith, and Justesen (1985) constructed a device for detecting rodent activity in which 8 phototransistors and corresponding IR LEDs were aligned horizontally. Data were recorded and stored using Labsoft and Applesoft programming. Wilson (2004) described how an IR LED and a photo Darlington (i.e., an array of two transistors and a resistor), could be used in inexpensive photocells that work in combination with MED Associates and Coulbourn commercial equipment. As noted by Wilson, because photo Darlingtons are affected by changes in illumination, each photo Darlington must be placed inside an opaque tube. Similarly, when phototransistors are used, it is necessary to calibrate hardware carefully; for example, by varying the value of resistors until the phototransistor functions properly.

The photocells described in this paper use inexpensive IR LEDs (Model IR383) and phototransistors (Model PT1302B / C2), and work in combination with the Arduino-Visual Basic interface using programs similar to those described by Escobar and Pérez-Herrera (2015). One pair of phototransistor / IR LEDs are approximately 1 USD and can be acquired in almost any electronics store. To make the photocells suitable for diverse conditions of illumination and distances between the IR LED and the phototransistor, software calibration, rather than modifications of the physical arrangement of photocell, is required.

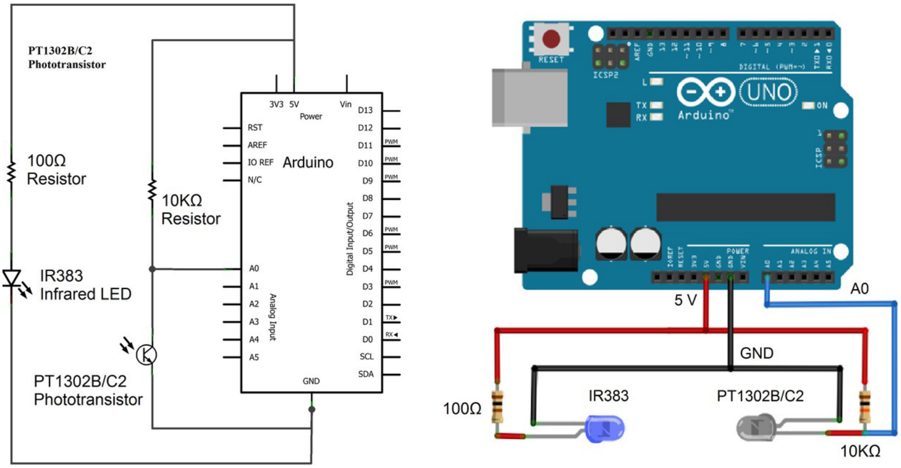


Figure 2. The electronic circuit connecting the phototransistor and an IR LED to an Arduino Uno board is shown in A. The physical arrangement is shown in B. An Arduino UNO board is used in the example but other Arduino boards (e.g., Arduino Mega), could be used.

Hardware Connection

Section A of Figure 2 shows the electronic circuit and Section B shows the physical connections using an Arduino UNO board. Phototransistor PT1302B / C2 is connected to one of the pins labeled A0 to A5 in Arduino UNO boards. These pins are known as analog and are equipped with an integrated analog-to-digital converter (ADC) that transforms a continuous voltage (e.g., 5 to 0 V) into discrete numbers. For example, as the voltage received by the input in the Arduino board varies depending on the IR light reaching the phototransistor from the IR LED, a 10-bit value ranging from 0 to 1023 is read as the input value. It is because input values vary from 0 to 1023 that these inputs are known as analog. To transform this analog value into response occurrences, a threshold value associated with light-beam interruptions must be calculated. This procedure will be described in the following section. The negative end (long terminal) of the phototransistor is connected to ground (pin GND) in the Arduino board, and the positive end (short terminal) is connected to pin 5 V in the Arduino board using a 10-Kohm resistor.

The positive end (long terminal) of the IR LED is connected to 5 V pin in the Arduino board using a 100-ohm resistor. The negative end (short terminal) is connected to pin GND in the Arduino board. With this connection, the IR LED is on whenever the Arduino board is powered[1]. Note that the long terminal of the phototransistor and the short terminal of the IR LED are connected to ground. If the phototransistor and the LED are perfectly aligned, the interruption of the beam of light can be detected even if the phototransistor and the IR LED are separated up to 2 m. This distance is reduced if the alignment is not perfect. The phototransistor and the IR LED can be aligned by placing them in an enclosure (see for example the one at https://www.thingiverse.com/thing:2345646) with slots on opposite sides of the enclosure

Software

Calibration Procedure

After the phototransistor and the IR LED are placed in the experimental setting, the values in the analog input associated with the interruption of the light beam (response), relative to the value obtained when the IR light reaches the phototransistor, must be determined. To determine such values an Arduino program must be downloaded from: http://analisisdelaconducta.net/wp-content/uploads/2015/10/Arduino_Photocell_Test.zip This program must be uploaded to the Arduino board using the Arduino Integrated Development Environment (IDE). Steps for uploading the program are described in Table 1.

To help users obtaining the threshold value for each phototransistor, we created a Windows program that determines the upper and lower limits of the analog value when the light from the IR LED is blocked and unblocked. This program can be downloaded from: http://analisisdelaconducta.net/wp-content/uploads/2015/10/Photocell_Calibration.zip. To run the program, extract the file "Photocell_Calibration.exe" and double click on it. The VCP assigned to the Arduino board (e.g., COM3), and the analog pin number (e.g., A0) in which the phototransistor was connected, must be entered in the program before starting the test. Afterwards, pressing the start button analyzes the upper limit and lower limit of the sensor values when the light reaches the phototransistor without interruption during 5 s. The values we observed in one test, with the IR LED and the phototransistor located at 10 cm from each other, ranged from 0

[1] An IR LED can also be connected to a digital pin in the Arduino board to turn it on and off as needed.

Table 1

Steps for running Arduino and Visual Basic programs

Step	Instructions
Uploading Arduino Programs	
1	Download Arduino IDE from http://arduino.cc/en/Main/Software and install it in a PC.
2	Connect the Arduino board to the USB port and verify the assigned COM port. This can be done by clicking "Tools" and "Port" in the Arduino IDE. The COM port used by the Arduino board will be followed by the legend "Arduino UNO". The COM port can also be verified using the "Device Manager" in the "Control Panel".
3	Open the Arduino program (.ino extension) using Arduino IDE and click the "upload" button to transfer the program to the Arduino board. Before uploading, check board model by clicking "Tools" and then "Board". Verify that the COM port selected in "Serial Port" also in "Tools", is followed by the label Arduino UNO.
Running Visual Basic programs	
1	Create folder "Data" in C:\.
2	Download the Visual Basic program from the authors' website.
3	Extract the file to C:\ …Documents\Visual Studio 2010\Projects. This will create folder Visual_Basic_Program. Inside the folder, double click on "Arduino_VB.sln" to open the program.
4	Execute the program by pressing "F5" or by clicking the "start debugging" icon. Introduce the values, select the schedule and click "continue". Confirm the data entered by clicking "continue".

to 30 (Min value = 30). After this first test is conducted, the program prompts the user to interrupt the light from the IR LED (enter the response) and press the start button to calculate the upper limit and lower limit of the sensor values when the light is blocked. In our test, for example, we observed values that varied between 970 to 990 (Max value = 970). When the two ranges of values are obtained, the program displays a value that is used as threshold. This threshold is calculated with the formula: Min value + ((Max value – Min Value) / 2). In our test this value was 500.

Using the Photocell with the Arduino-Visual Basic Interface

Minor modifications of the Arduino programs provided by Escobar and Pérez-Herrera (2015) are required to use photocells. The rationale is to consider that the target behavior occurs only when the value read is above the threshold obtained in the calibration process.

The Arduino program designed to be used with photocells, can be freely downloaded from the authors' website. If a single operant is to be recorded, download the program at the following link http://analisisdelaconducta.net/wp-content/uploads/2015/10/Arduino_Program_Photocell.zip. If two operants are to be recorded concurrently, download the program at the following link http://analisisdelaconducta.net/wp-content/uploads/2015/10/Arduino_Program_Photocell_Concurrent.zip. In each program the threshold value stored in the variables *Response_upper_limit* = XXX must be replaced with the value obtained with the calibration program. After the value is changed, the selected Arduino program must be uploaded to the Arduino board.

As noted by Escobar and Pérez-Herrera (2015), Visual Basic programs are necessary to generate schedules of reinforcement. These programs written in Visual Basic 2010 Express Edition can be downloaded for free. These programs are the same described by Escobar and Pérez-Herrera. One program generates simple schedules of reinforcement when a single photocell is used (http://analisisdelaconducta.net/wp-content/uploads/2015/04/Visual_Basic_Program.zip) and must be used in combination with the Arduino program for one operant. The second version generates concurrent schedules of reinforcement when two photocells are used (http://analisisdelaconducta.net/wp-content/uploads/2015/03/Visual_Basic_Program_Concurrent.zip) and must be used in combination with the Arduino program for two concurrent operants. Instructions for running the programs are included in the lower section of Table 1. Whenever the program in Visual Basic generates experimental events and reads the state of inputs it writes a code for the event with a time stamp in a .txt file. Connecting devices used in operant conditioning chambers, which generally require 28 V DC, requires adding an array of relays and a power supply (see Escobar & Pérez-Herrera, 2015, p. 430).

Although the system described can be used with one or two photocells, it can be modified to add up to 6 photocells using the 6 analog pins of the Arduino UNO board. Two or more IR LED can be connected in parallel to the Arduino board (i.e., each connected to GND and to 5-V using a 100-ohm resistor). Each IR LED consumes approximately 20 mA of the 500 mA provided by the USB port through the 5-V pin of Arduino boards. When photocells are used with the Arduino-Visual Basic interface responses of a minimum duration of 5 ms and a minimum interresponse time also of 5 ms, are accurately recorded, which is a maximum resolution of

100 responses per second. These values were tested by turning on and off repeatedly an IR LED with an independent Arduino board. Simulated response durations and interresponse times were verified with an oscilloscope. It is important to note that Visual Basic's timer has a 15 ms-resolution; therefore, two or more responses occurring within a 15 ms-interval are recorded with the same time stamp. As noted by Escobar and Pérez-Herrera (2015), programs intensively accessing the computer's hard drive (e.g., antivirus software) could affect performance and must be closed while experimental sessions are running.

The programs described in this paper that are used to record responses using IR LEDs and phototransistors also can be used for recording discrete responses with other analog sensors. For example, by removing the IR LED and replacing the phototransistor with a force sensor (e.g., Interlink, Model FSR 402 force sensing resistor) a response can be detected whenever the sensor is pressed. Also, a flexion sensor (e.g., Sprectrasymbol, Model FSL0055253ST) could be used to detect a response when the sensor is bent.

References

Antonitis, J. J. (1951). Response variability in the white rat during conditioning, extinction, and reconditioning. *Journal of Experimental Psychology, 42,* 273-281.

Batson, J. D., & Turner, J. D. (1986). A simple infrared photocell device and its interface to an Apple computer. *Behavior Research Methods, Instruments & Computers, 18(5),* 447-451.

Clarke, R. L, Smith, R. F., & Justesen, D. R. (1985). An infrared device for detecting locomotor activity. *Behavior Reserch Methods, Instruments, & Computers, 17 (5),* 519-525.

Dixon, M. R., & MacLin, O. H. (2003). *Visual Basic for behavioral psychologists.* Reno, NV: Context.

Ericson, E., Samuelsson, J., & Ahlenius, S. (1991). Photocell measurements of rat motor activity. A contribution to sensitivity and variation in behavioral observations. *Journal of Pharmacological Methods, 25,* 111–122.

Escobar, R. (2014). From relays to microcontrollers: The adoption of technology in operant research. *Mexican Journal of Behavior Analysis, 40*(2), 127-153.

Escobar, R., & Pérez-Herrera, C. A. (2015). Low-cost USB interface for operant research using Arduino and Visual Basic. *Journal of the Experimental Analysis of Behavior, 103,* 427-435.

Lattal K. A., & Metzger, B. (1994). Response acquisition by Siamese fighting fish *(Betta splendens)* with delayed visual reinforcement. *Journal of the Experimental Analysis of Behavior,* 61, 35-44.

Manabe, K. (2017). The Skinner box evolving to detect movement and vocalization. *Mexican Journal of Behavior Analysis,* 43, 192-211.

Muller, P. G., & Cheney, C. D. (1975). The automatic recording of aggression in pigeons: A comparison of two methods with live targets. *Behavior Research Methods* & Instrumentation, 7, 357-360.

Varnon, C., & Abramson, C. I. (2013). The propeller experiment controller: low-cost automation for classroom experiments in learning and behavior. *Innovative Teaching,* 2(2), 1-18.

Wilson, W. J. (2004). An alternative to commercial infrared sensors. *Journal of Undergraduate Neuroscience Education,* 3(1), A1-A3.

Recibido Junio 29, 2017 /
Received June 29, 2017
Aceptado Agosto 7, 2017 /
Accepted August 7, 2017