MATHEMATICS AND ROBOTS

Matthew D. Mogensen[1]
Department of Mathematical Sciences
United States Military Academy
West Point, NY 10996
matthew.mogensen@usma.edu

Robots have been used in education for many decades. For example, the Turtle was an educational robot produced by Valiant Design starting in the 1980's.

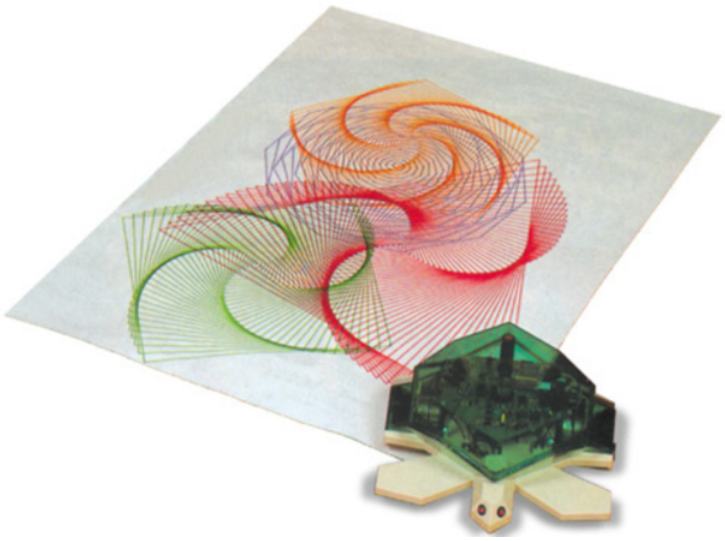

Figure 1: The Turtle, produced by Valiant Design

Robots can be used to to further an interest in Science, technology, Engineering and Mathematics (STEM), and if used effectively in the classroom, can engage students in the learning process. The projects discussed here rely on the Arduino ecosystem, which is "an open-source physical computing platform based on a simple microcontroller board, and a development environment for writing software for the board"[2]. While there are many other software and computing platforms available, the large open-source community of developers and hobbyists is constantly increasing the availability of new projects and hardware add-ons.

[1]The views expressed herein are those of the author and do not reflect the position of the United States Government, the United States Army, or the United States Military Academy.

[2]http://www.arduino.cc/en/Guide/Introduction

Follower Robot Project

We strongly recommend first working through the online documentation for the Parallax Robot Kit as well as the CMUCam5. This project is a basic robot with a fixed position camera in front.

Supplies:

1. Parallax BOE Robot ($125)
2. Arduino Uno Board ($25)
3. CMU5 Pixy Camera with included cable ($69)
4. 6 wires

Instructions:

The camera comes with a cord that is specifically designed to be connected directly into the camera while the other side goes directly into the specified port on the Arduino (following the CMUcam5 documentation[3]). There is only one way that this wire can be connected to the board and camera (Figure 2).

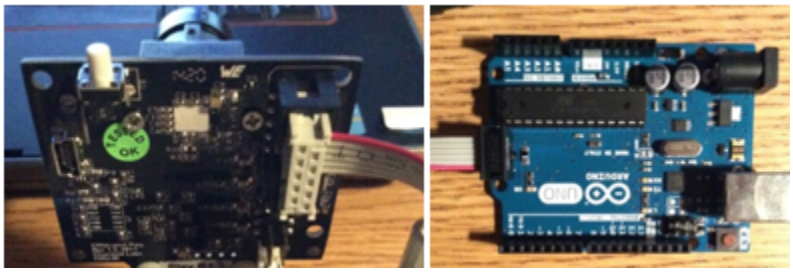

Figure 2: Connecting the camera to the Uno

The Parallax BOE Robot is designed to connect to an Arduino Uno. Once the camera is connected to the Uno as described above, slide the Uno onto the pins on the underside of the Parallax board. Unscrewing the top board (four screws - one in each corner) makes this task much easier and is highly recommended (Figure 3).

[3]http://cmucam.org/projects/cmucam5/wiki/Hooking_up_Pixy_to_a_Microcontroller_like_an_Arduino

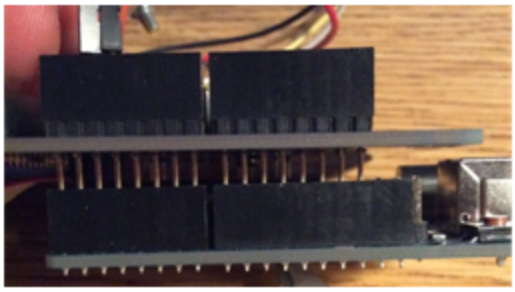

Figure 3: Connecting the robot chassis to the Uno

There are two sets of three wires (red, white, black) that come from the wheel motors. Each will have a similar wiring to to connect them to the Robot (Figure 4). The white wire of the left servo goes to pin 3 and the white wire of the right servo goes to pin 2. The red wires go to the 5v power. The black wires goes to ground. In order to connect the wires to the proper pin hole, it is necessary to use an additional wire to go from the mouth of the servo wire into the proper pin hole. These wires can be found in an Arduino kit. Any wire that can connect the servo to the proper pin hole will work. The white wires determine which pin the Uno will use to send commands to the wheel motors. If your wheel motors turn in reverse of what you expect, be sure to check pins 2 and 3 to make sure your code reflects which one is wired to each wheel motor.

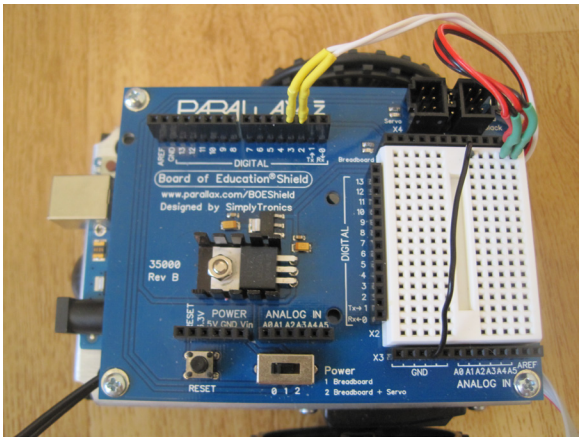

Figure 4: Connecting the wheel servos wires to the Uno

The next step is to "train" the Pixy camera to recognize objects of a certain color. The Pixymon software and software instructions and should first be downloaded and consulted.[4]

[4]http://www.cmucam.org

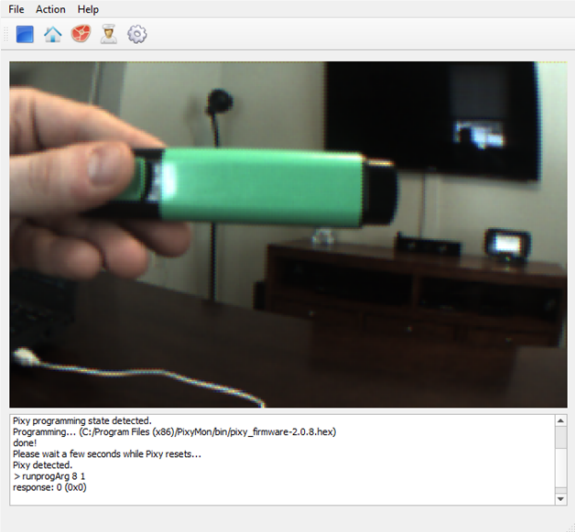

Figure 5: View from camera in Pixymon program

You should set the desired object in front of the camera and select the option "Set Signature #1" under the "action" tab. Keep in mind that the Pixy camera will want to track any object that has the same color, therefore an object whose color is different than the environment in which the robot will be used is preferred. Ensure the lighting is constant and also reflects the environment in which the robot will be used. The user will select where in the picture his object is, and the camera will then remember this color (until it is reset via the Pixymon software or the hardware reset button).

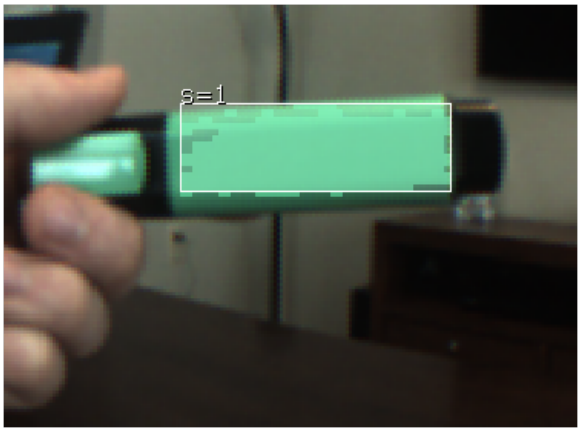

Figure 6: Acquiring an object's color in Piximon software

Once the camera is "trained" to recognize the desired color, attach the camera to the front of the robot with extra screws included in the Parallax Robot kit (Figure 7).

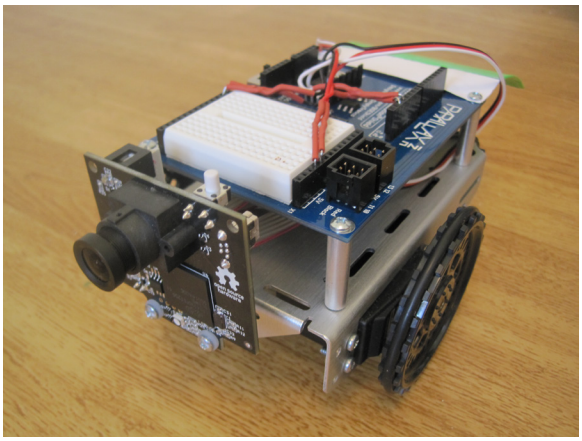

Figure 7: The camera attached to a robot frame

Download the file *robot.ino* available at http://GOO.GL/L2JIIX. Open the file with the Arduino IDE[5]. Connect the computer via USB to the Uno and upload the program, then turn the robot on.

Discussion of Algorithms

The secret to improving the "way" your robot follows an object lies in mathematics. This gives educators important talking points to expand in different areas of mathematics, including the concepts of constants, ratios, derivatives and integrals. It is important to note that a proper following algorithm must consider not only which way to turn (left or right), but also what the speed of each servo should be, whether the robot should come to a stop, speed up, or slow down. There are many different algorithms that can be easily coded using the Servo and Pixy libraries.

Proportional-Derivative-Integral (PID) control, used widely in the field of industrial control[6], can motivate different approaches. PID control is a feedback loop algorithm implemented in software or hardware, which allows an output to be adjusted based on current or previous feedback. The output is a sum of three terms, a proportional term, a derivative term, and an integral term, shown in order:

$$u(t) = K_p e(t) + K_d \frac{d}{dt} e(t) + K_i \int_0^t e(\tau) d\tau$$

$u(t)$: output (how much to turn each wheel servo)
K_p, K_d, K_i : tuning parameters
e : error (this is the angle between the robot camera and the object to follow)
t : present time
τ : variable of integration (time 0 to present)

[5]available for free at www.arduino.cc
[6]http://www.ni.com/white-paper/3782/en/

For the robot algorithm, we can capture the horizontal error as the distance from the center of the object to the camera's center field of view, and the distance error (how far away the robot is from the object) as a function of the size of the object in the camera's view. The proportional term adjusts the output based on the current error. The derivative term contributes to the output based on the rate of change of the error, and the integral term contributes to the output based on the total error from the beginning (smoothing term).

One or more of these terms can be incorporated in a feedback control mechanism (Figure 8)[7] that controls the robot's wheel servos.

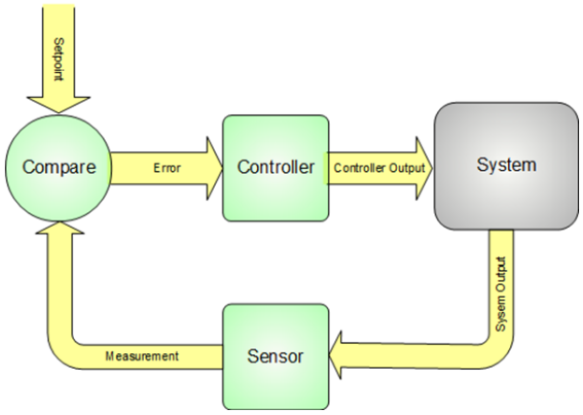

Figure 8: Feedback control diagram

On-Off Control:

An On-Off control algorithm calculates an output to send to the wheel servos from a discrete set of outputs, dependent on the error.

- Horizontal alignment: If the object being tracked is left of center, turn left at a constant rate. If the object being tracked is right of center, turn right at a constant rate.
- Robot speed: If the size of the object being tracked is smaller than a preset tolerance, then move at a constant speed until you reach a certain distance from the object, then stop.

[7]Feedback Control Diagram from learn.adafruit.com, accessed April 2015

Proportional (P) Control:

A proportional control algorithm calculates an output value that is proportional to the magnitude of the error.

- Horizontal alignment: If the object being tracked is left or right of center, turn left or right at a rate proportional to the error.
- Robot speed: If the size of the object being tracked is smaller than a preset tolerance (the object is far away), move forward at a rate proportional to the distance from the object. If the size of the object being tracked is larger than a preset tolerance (you are too close), stop forward movement.

Proportional-Derivative (PD) Control:

A proportional-derivative control algorithm uses a weighted response of proportional and derivative terms in its output. Since the derivative term takes into account the rate of change of the error, an error that is rapidly approaching zero will decrease the output in order to "avoid overshooting". In the same manner, an error that is increasing will cause the output to increase to "catch up". There are many variations for controlling horizontal alignment and speed using PD control.

Advanced Project

Projects using pan-tilt servos[8] ($19) for the camera (Figure 9) can be use more advanced algorithms. A similar project by Adafruit built on the Zumo Robot chassis[9] is a good resource for open-source code using PD control feedback in its pan-tilt algorithm. The Arduino program *pantiltrobot.ino*, based heavily on the Adafruit pixy pet follower robot project[10], but modified to work with the Parallax BOE chassis can be downloaded at http://GOO.GL/L2JIIX.

[8] http://www.adafruit.com/product/1967
[9] http://www.adafruit.com/product/1639
[10] https://learn.adafruit.com/pixy-pet-robot-color-vision-follower-using-pixycam

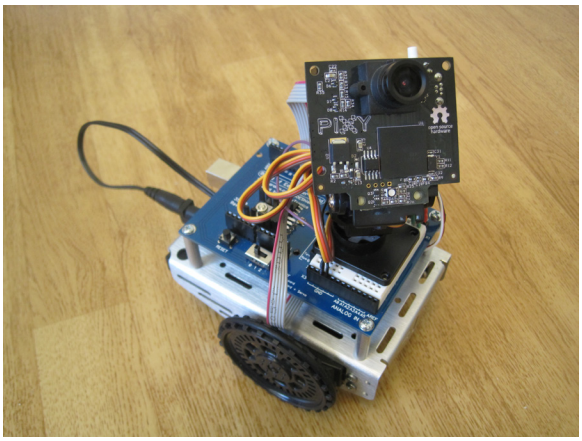

Figure 9: Follower robot with pan-tilt camera servos

In the Classroom

While the current robot provides a simple "on-off" following algorithm, students can develop more sophisticated algorithms quite easily by incorporating several mathematical concepts learned from PID feedback control. The mathematics that can be explored with various algorithms serve as a gateway to classroom projects, increased STEM interest, and student independent study.

References

[1] Computer History Museum,
"http://www.computerhistory.org/collections/catalog/102662778, 2015.

[2] Arduino, http://www.arduino.cc/en/Guide/Introduction, 2015.

[3] Parallax, Inc. https://www.parallax.com/product/boe-bot-robot.

[4] Carnegie Mellon University, http://www.cmucam.org/.

[5] Adafruit, https://blog.adafruit.com/2014/08/26/pixy-pet-robot-color-vision-follower-the-adafruit-learning-system/.

[6] National Instruments (NI), http://www.ni.com/white-paper/3782/en/