Lesson

EVO VACUUM SIMULATION

CREATED BY

Richard Born

Associate Professor Emeritus
Northern Illinois University
rborn2@niu.edu

TOPICS

Robotics, Computer Science, Programming

AGES

Grades 7-12

METHOD

OzoBlockly

DURATION

30-40 Minutes

OzoBlockly Lesson:
EvoVac—Robotic Vacuum Cleaner Simulation

By Richard Born
Associate Professor Emeritus
Northern Illinois University
rborn2@niu.edu

Introduction

The idea of not having to clean floors and carpets in your home is apparently quite luring to consumers. With dozens of robotic vacuum cleaner manufacturers out there, customers certainly need to do their homework before purchasing. To mention just a few, you can buy Black+Decker, bObiPet, Dyson 360 Eye, Ecovacs Deebot, Hoover Quest, iLife A4, Metapo Infinuvo, iRobot Roomba, iTouchless, LG Hom-Bot Square, Neato Botvac, Rollibot BL618, Samsung POWERbot, and V-Bot P3. Prices vary from a couple hundred dollars to more than a kilobuck. Features may include remote control from your iPhone or Android, automatic recharging from a docking base, the ability to pick up pet hair and clean paw prints, UV disinfection, stair detection to avoid falling, staying within preset boundaries, maneuverability under and around furniture, scheduling, and simultaneous vacuuming/mopping/sweeping.

The robotic vacuum cleaner is just one example of the proliferation of robots for a large variety of functions in modern society. But it does help to point out the need for robotic engineers and programmers, as well as the importance of teaching robotics in the classroom. In this lesson, students will study and run an OzoBlockly program for which Evo simulates a robotic vacuum cleaner. No, Evo will not actually be picking up dust while moving, as this would certainly clog his wheel mechanism. Nor will he be simulating all of the features discussed in the previous paragraph.

We'll have Evo maneuvering a rectangular room in the form of a small cardboard box, avoiding walls as he moves back-and-forth across the room. We'll also make Evo put his IR sensors to use by helping him turn himself when he approaches a wall to make the next swipe of the room. In the process of doing this lesson, students will gain experience in working with blocks including *if…else*, loops, several *free movement* blocks, and the *read proximity sensor* block.

See Figure 1(a), which diagrams our plan on how Evo is to move, starting in the lower left corner of the room and traversing the room as shown by the arrowed line segments. Figure 1(b) is a photo showing Evo at the start location in the room, represented by a cardboard box 7" x 11" in size. Figure 1(c) is a long exposure photograph showing the trail of Evo's top light as he "vacuums" the box. We want the top light to be green while Evo travels unimpeded, blue as he nears a wall, and red as he turns around for the next swipe of the room.

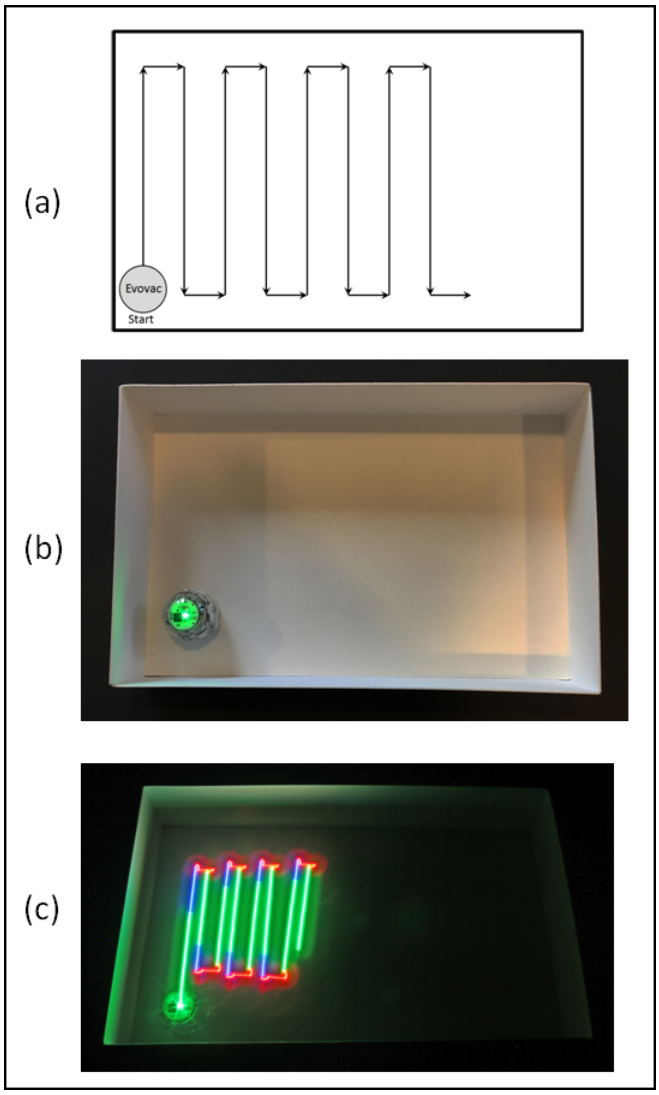

Figure 1

Figure 1(c) shows that there is some limitation to the rotation precision of Evo. As pointed out in the online *OzoBlockly Reference*, this is due to a number of factors including traction differences between different operating surfaces (wheel slippage), inertia, speed calibration, and hardware design limitations. However, the user can tune some for these limitations by adjusting parameters of the move and rotate blocks.

You are encouraged to view the short video (https://www.youtube.com/watch?v=5XbW76gd5So) that accompanies this document. Doing so will provide a better feeling for what our OzoBlockly program does.

The OzoBlockly Program

Figure 2 shows the blocks making up the OzoBlockly program that allows Evo to simulate a simple robotic vacuum cleaner. The discussion that follows will detail the purpose of the blocks, some being straight forward and others more subtle in what they accomplish.

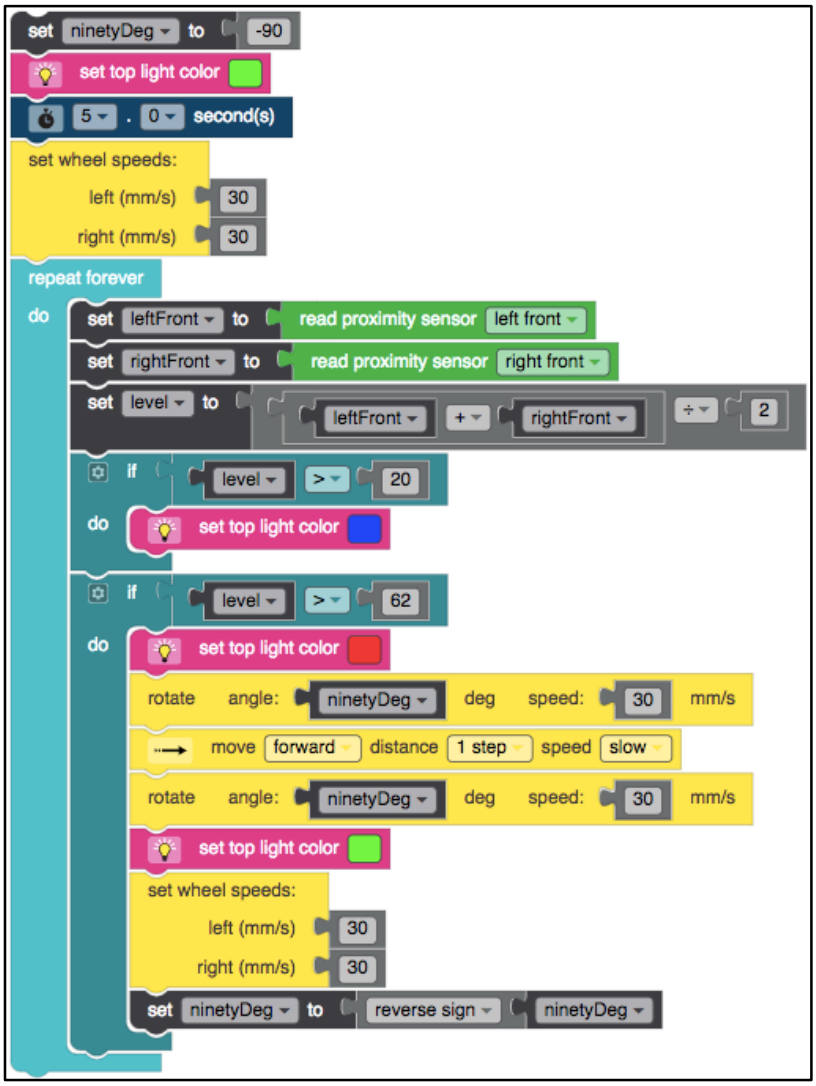

Figure 2

First, let's discuss the variable *ninetyDeg* (*Deg* meaning "Degrees"), which is initialized at program start to the value -90 and the yellow rotate and move blocks near the center of the program. As indicated in the online *OzoBlockly Reference*, negative rotations turn Evo to the right, while positive rotations turn Evo to the left. Since we are starting Evo in the lower left corner of the box, when he reaches the top he needs to turn right, then take a step forward toward the right side of the box, and then make another right turn so that he is facing the bottom of the box. This explains why the variable *ninetyDeg* is initialized as negative. When Evo reaches the bottom of the box, he needs to repeat this process by making left turns. This is accomplished by reversing the sign of the variable *ninetyDeg*, which occurs at the end of the *do* portion of the *if…else* block.

The top LED is initially set to green and then the user is given 5 seconds to place Evo in the lower left corner of the box facing toward the top of the box. We want Evo to execute free movement, so the wheels speeds are both set to 30 mm/s. The rest of the program consists of a *repeat forever* loop in which Evo simulates the vacuuming of the room. To keep things simpler, we're not going to worry about what Evo needs to do when finishing vacuuming. The user will simply lift Evo out of the box and press the button once to turn Evo off.

Now let's discuss what happens inside of the *repeat forever* loop. The left and right sensors are read, and the average of these two readings is stored in the variable *level*. According to the online OzoBlockly Reference, level will turn out to be a number between 0 and 127, with 0 indicating no obstacle is detected, and increasing values indicating that Evo is getting closer and closer to the wall it is approaching.

Now let's discuss how to decide when the top LED is green, blue, or red. We want it green when Evo is moving unimpeded, blue when it is getting close to a wall and red when it is turning around. We found, again through trial and error, that if the average IR sensor level is greater than 20, Evo was getting relatively close to the wall, so in that case we set the top LED to blue. If the level is greater than 62, Evo was close enough to a wall that it was time to turn around. In that case we set the top LED to red. After turning around, the top LED is again set to green as Evo is moving unimpeded toward a wall.

Student Activities

1. After viewing the video (https://www.youtube.com/watch?v=5XbW76gd5So) and discussing the OzoBlockly program, motivate the students by letting them load, run and play with the program *EvoVacuumSimulation.ozocode*.
2. Then ask the class to modify the program by performing the following program maintenance task:

    In the program that was discussed in class, we didn't consider what Evo needs to do when finishing vacuuming. We just had an infinite loop and stopped Evo by pressing his button. Modify the program so that EvoVac stops after making 10 swipes of the room. Do this by the use of a *do while* loop.