Evo OzoBlockly Challenge: Random North-South-East-West Walk

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

One of the most famous random walk problems involves a person, who with each step, moves one distance unit North, South, East, or West with random, but equal probabilities. You can think of the person as moving about on an infinite two-dimensional Cartesian coordinate plane. If he takes a step north, y is increased by 1; if he steps south, y is decreased by 1. If he takes a step east, x is increased by 1; if he steps west, x is decreased by 1.

Programming Ozobot to do this and keep track of his current coordinates is quite challenging, as Ozobot's innate direction moves are not North, South, East, and West. Rather, they are left, right, straight, and back, as shown in Figure 1.

Figure 1

What we need are four functions (or procedures), and we might name them **goNorth**, **goSouth**, **goEast**, and **goWest**. We would also make use of a variable, maybe called *curDir*, which would keep track of the current direction that *Evo's leading edge is facing*. Figure 2 shows a possible way to assign values to the variable *curDir*: 0 if Ozobot is facing north, 1 if he is facing east, 2 if he is facing south, and 3 if he is facing west.

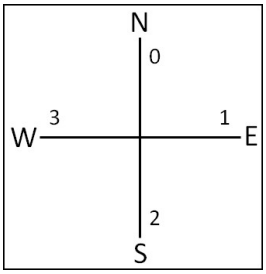


Figure 2

The basic skeleton structure of the random walk then might appear as shown in Figure 3. The four functions **goNorth**, **goEast**, **goSouth**, and **goWest** are randomly called in response to the value of a random number between 1 and 4 inclusive. The job of each of the four functions is then two-fold:

- Depending on the value of *curDir*, pick direction left, right, straight or back, and update the value of *curDir*. For example, suppose that *curDir* is 2 and **goWest** has been called. With a current direction of 2, Ozobot would be facing south. So to go west, he would need to pick direction right and update *curDir* to 3.
- Update the *x* or *y* coordinate of Evo. For example, **goNorth** would increase *y* by 1.

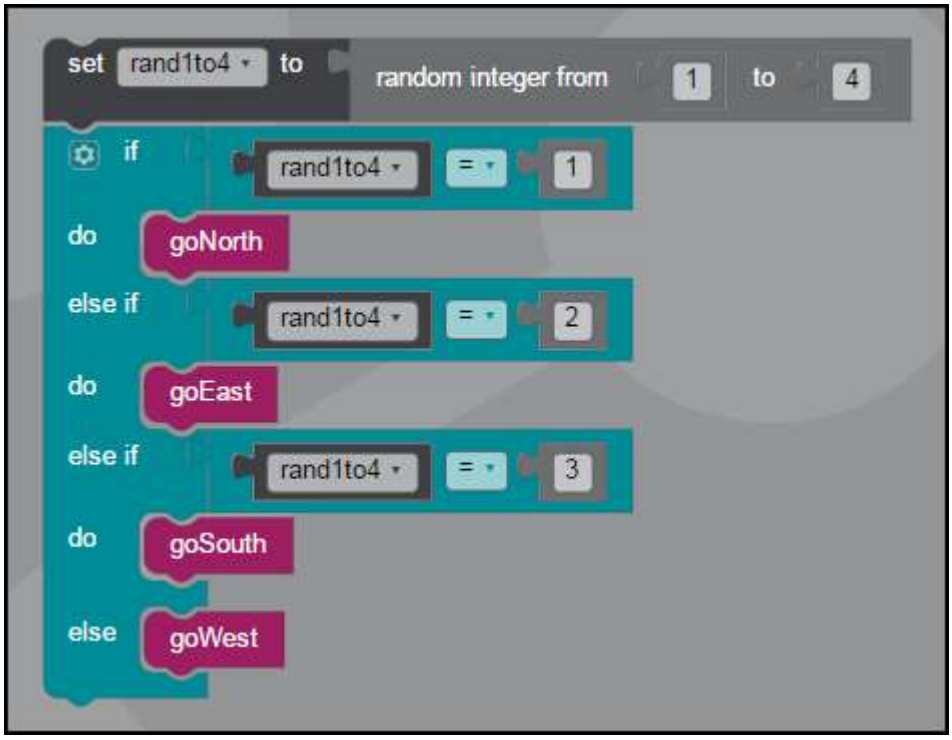


Figure 3

The Map for this Challenge Lesson

Remember that your program is to be written assuming that Ozobot has an infinite Cartesian plane to on which to move. Since an infinite plane is rather impossible for us to make, we will make use of a finite plane as shown in Figure 4. A larger version, suitable for actually running Evo, is found on the last page of this document. When Evo approaches the line ends at the top, bottom, left, and right of this map, your program will cease to run and Evo will stop and blink red/blue quickly. There is another file called *NSEW Large Cartesian Plane.pdf*. If you use this file to test/run your program, you will need to print a copy of the file and then take the copy to a copy shop and have them enlarge it from 8½" x 11" to 17" x 22", thus doubling the size. The advantage of the 17" x 22" grid is that Ozobot will be able to run your program for a much longer time before reaching the endpoints, stopping, and blinking red/blue. (The cost of having the enlargement made should be in the ballpark of $2.00, as it requires only black and white and not color printing.)

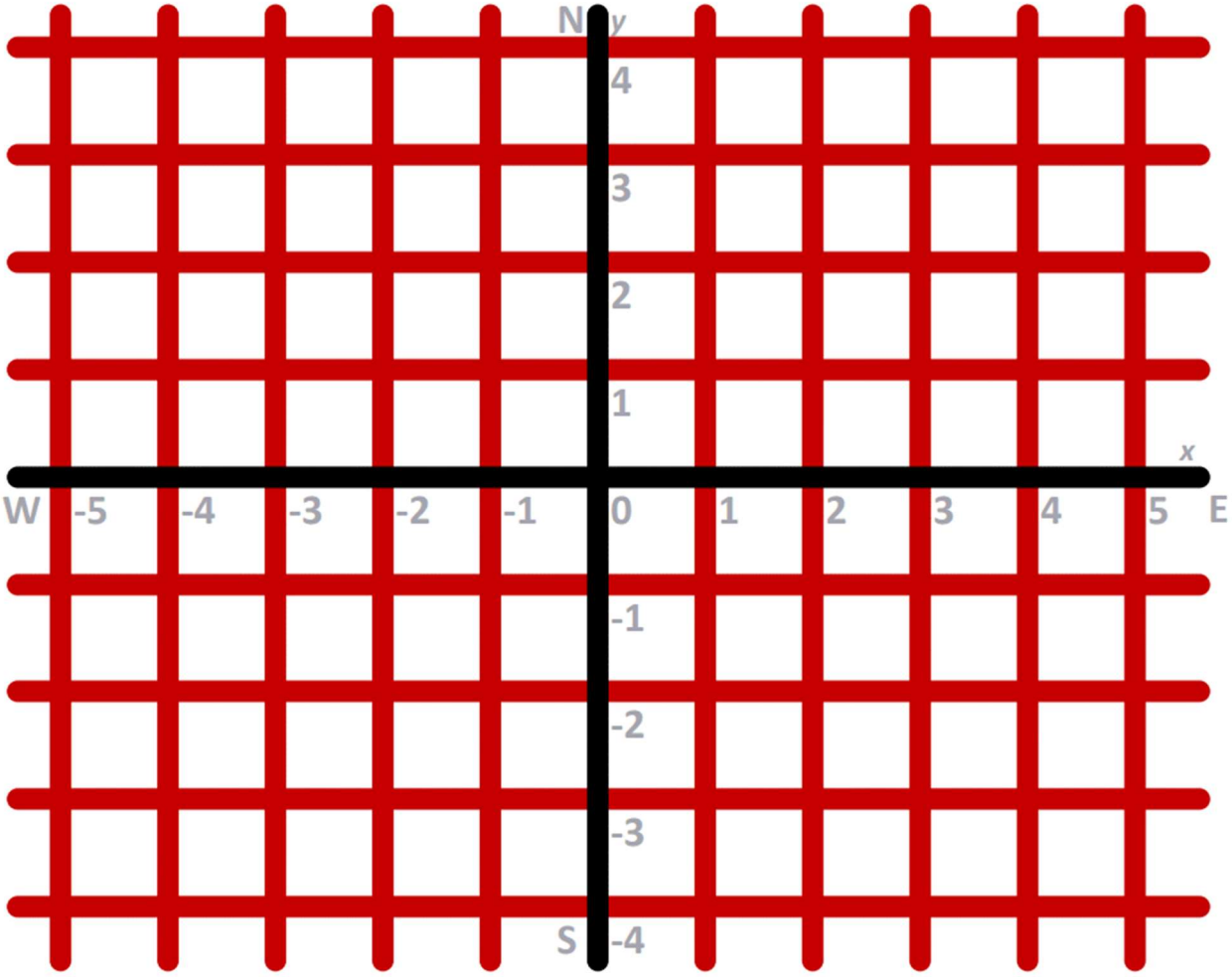
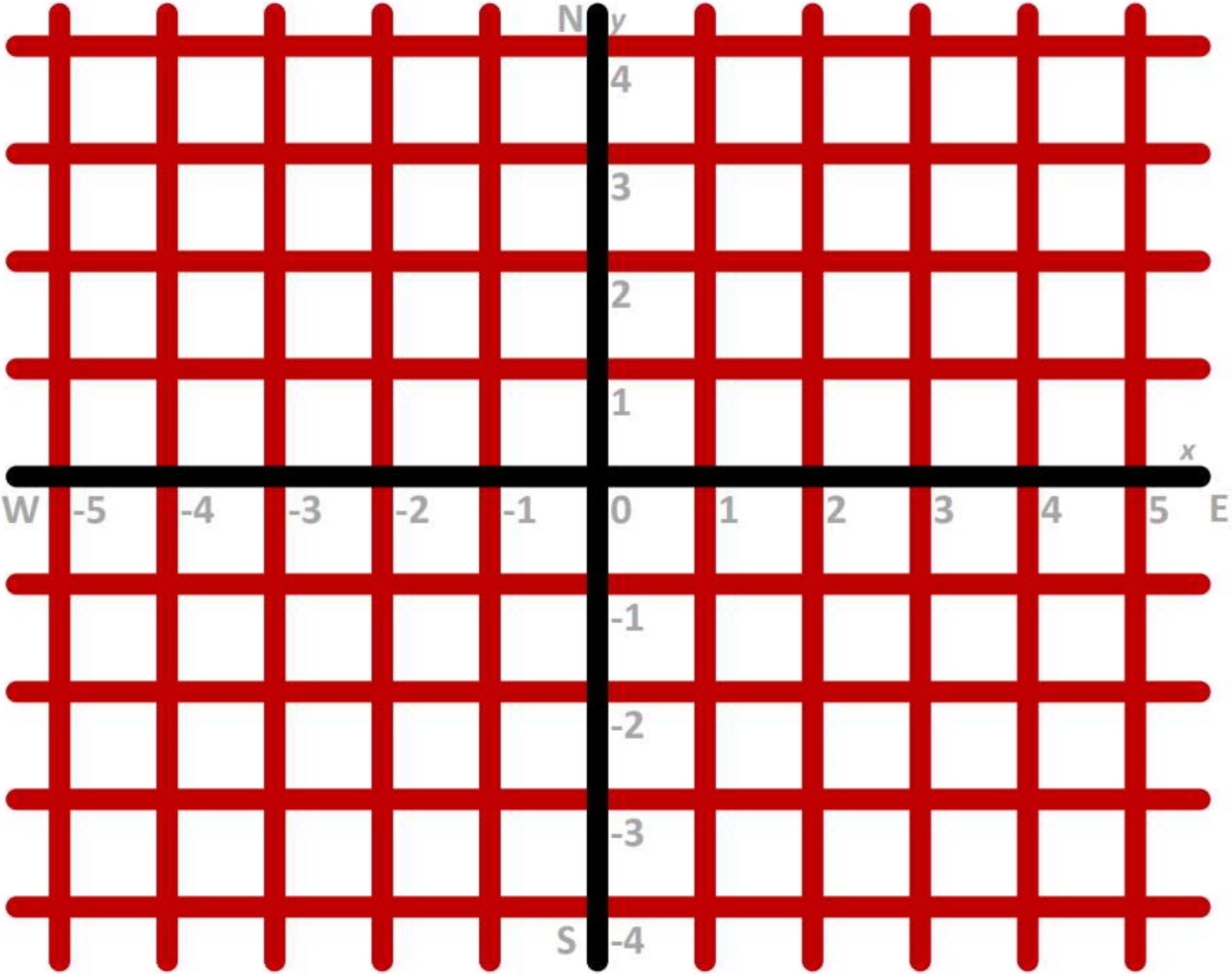


Figure 4

Challenge Requirements

1. While moving on the Cartesian place, Evo should display a white LED.
2. Evo should always be placed on the origin at (0, 0) facing north and then started. Ozobot's first move will be to go north to (0, 1). Therefore, you can initialize *curDir* at 0, *x* at 0, and *y* at 1. From there on, he will make random moves as described earlier.
3. Your first goal will be to write and test the main program and four functions **goNorth**, **goEast**, **goSouth**, and **goWest**. In order to determine if your program is working correctly, you need to have Evo display a specific color LED for one second, *indicating the direction that Ozobot is going to go next.* Display yellow for north, green for east, dark blue for south, and aqua for west.
4. Now you want to add code so that Evo speaks his coordinates *right after he has made each of his moves*. First speak the *x* coordinate, then the *y* coordinate.
5. It is rather time-consuming viewing coordinates after every move. Therefore, after successful completion of step 4, modify your program so that the coordinates will be displayed after every 5 steps.
6. If you want to disable showing coordinates all together, you simply need to right click on these blocks and select "Disable Block" from the drop-down menu.


N
y
4
3
2
1
x
W
-5
-4
-3
-2
-1
0
1
2
3
4
5
E
-1
-2
-3
S
-4