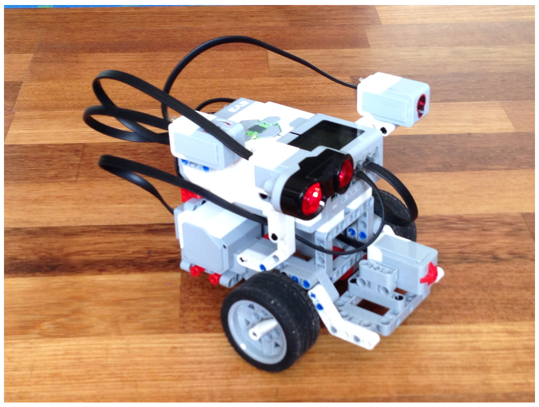
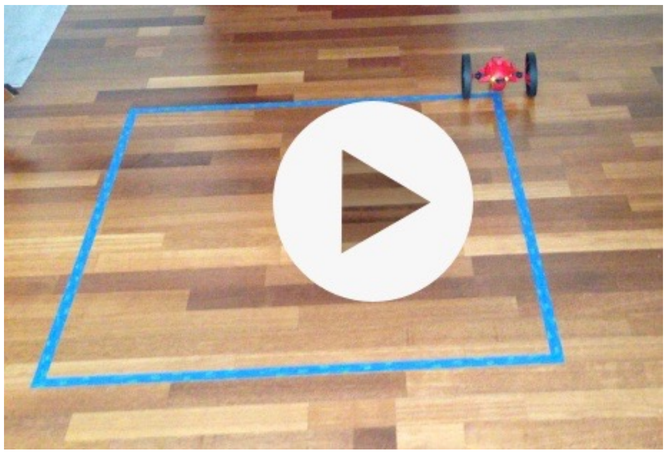
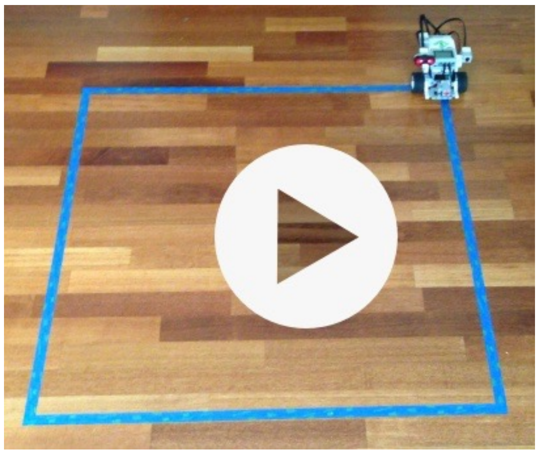
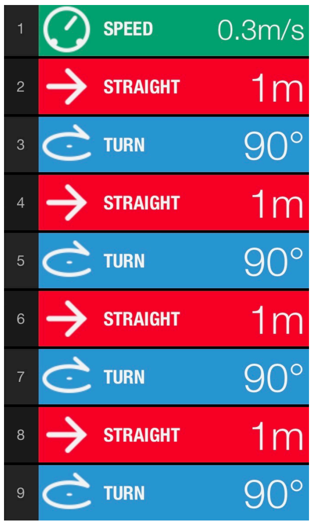
EV3 Code to Drive in a Square



I thought a good first step in learning about Lego Mindstorms EV3 was to build a Tribot and have it drive around a square 1m on edge. There is a plethora of information about EV3. Do a Google search on “build EV3 Tribot” or “EV3 drive in a square”. The number of sites identified is overwhelming and it’s difficult choosing where to start. The Art of Lego Mindstorms EV3 Programming, by Terry Griffen, is perfect as a getting started guide. I made some modifications to the build so I could include all 4 sensors that came with the kit. The 2 videos shown below show the Parrot drone and EV3 navigating a 1m square.



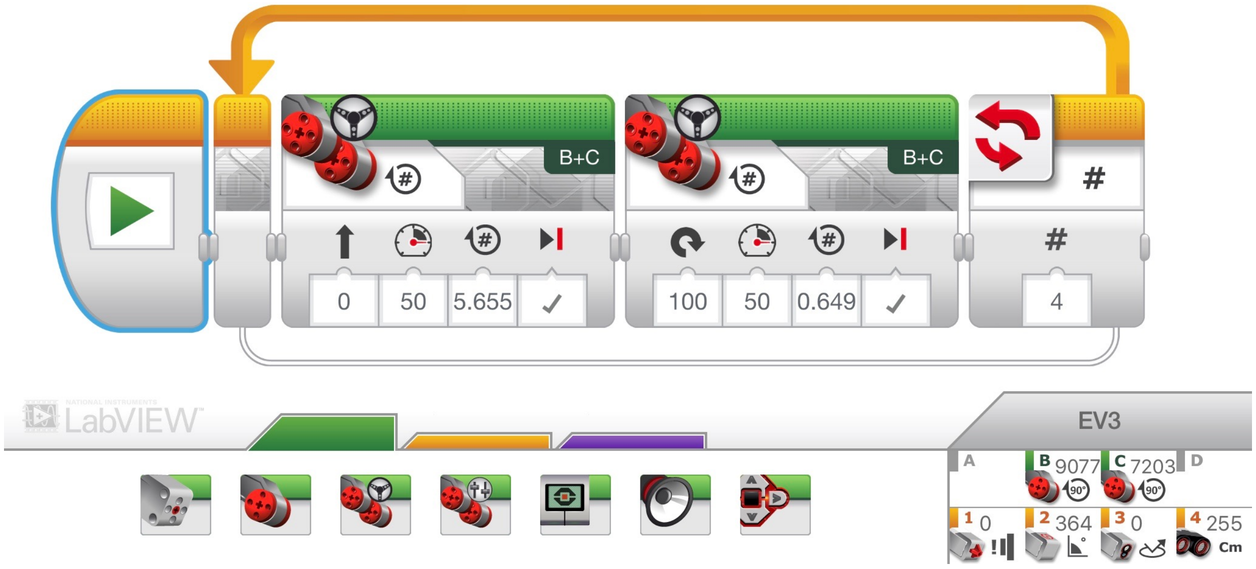




```
SPEED     0.3m/s
STRAIGHT  1m
TURN      90°
STRAIGHT  1m
TURN      90°
STRAIGHT  1m
TURN      90°
STRAIGHT  1m
TURN      90°
```

Writing the code to direct the Parrot drone to drive in a square is straightforward. The Parrot iPad app has the macro commands for “straight” and “turn” as shown in the code to the left. The only shortcoming is that “loops” are not available so the commands to drive straight 1m and turn right 90 degrees have to be repeated 4 times.

Code for EV3 is a bit more involved. There are several ways to write the code to direct EV3 to drive straight and turn. One way is shown in the following code. In the lower right corner, LabVIEW shows that the 2 motors are connected to ports B and C. These are then used in the



two “move-steering” command blocks. The sensors, which are not currently being used, are connected to ports 1, 2, 3, 4. The code consists of 2 “move-stearing” blocks contained within a loop to repeat 4 times. The move-steering command is like a LS-Dyna keyword. Four input parameters are required and their meaning changes with the mode chosen for the command by clicking the clockwise arrow in the upper left corner of the command. The first move-steering block to direct EV3 to move in a straight line for 1m requires 4 input parameters.

1. 0 == both motors synchronized to move EV3 in a straight line
2. 50 == motor power
3. 5.655 == number of motor rotations. The meaning of this parameter changes according to the mode selected (e.g., seconds, degrees, rotations).
4. brake at end

So, how did I know to enter 5.655 for the number of rotations. Several training examples suggest measuring how far EV3 travels for 1 rotation. Then divide this number into 1m to get the number of rotations needed. A more elegant way is to measure the wheel outer diameter and then hand calculate the rotations.

wheel d=0.05629m (using a Vernier caliper)
wheel circumference = πd = 0.1768 m/rotation
number of rotations = 1m / 0.1768 = 5.655 rotations

The second “move-steering” block to direct EV3 to turn right 90 degrees requires 4 input parameters.

1. 100 == one wheel turns clockwise and the other counterclockwise at the same rate resulting in EV3 rotating around a vertical centerline bisecting the wheel axel.
2. 50 == motor power
3. 0.649 == number of motor rotations
4. brake at end

The number of rotations was calculated as follows:

wheelbase = 0.1460m = rotational diameter
90 degree turn circumference = 0.1460 π / 4 = 0.1147
number of rotations = 0.1147 / 0.1768 = 0.649

The code block for the gyro sensor can be included in the code which allows direct entry of 90 degrees for the turn. My next step is to activate and play with the various sensors.