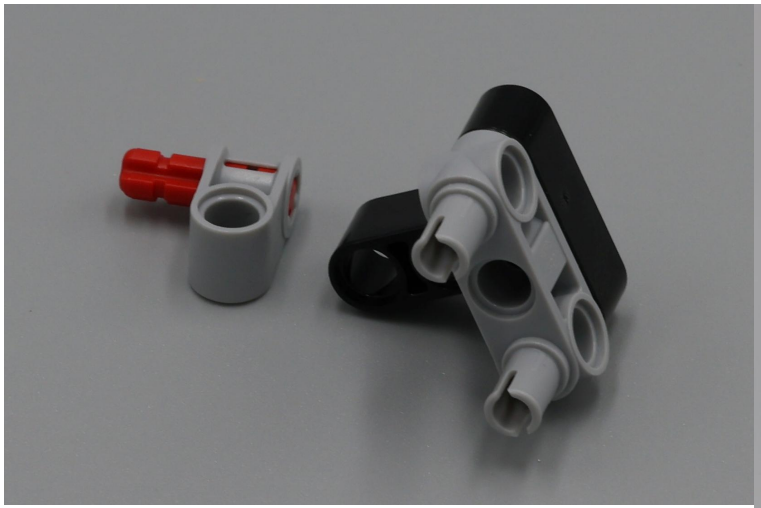
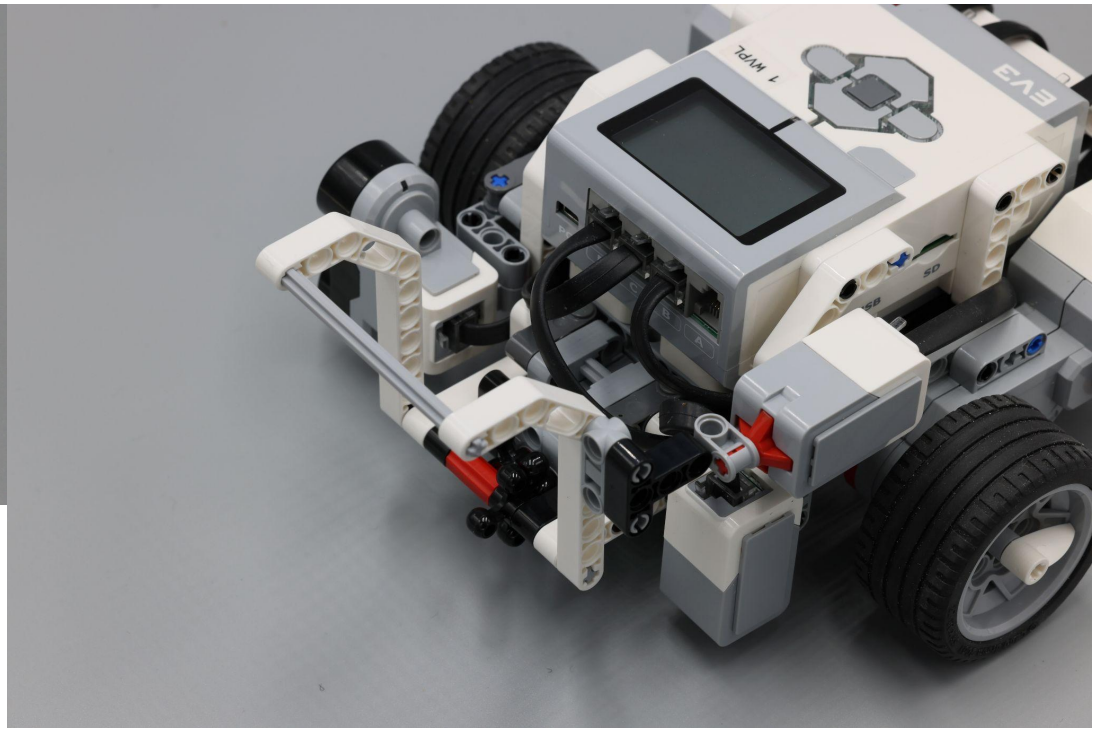
WVPL 2022 Makers Club Amazing Maze Lessons 3-5

Olavi Kamppari
Jan 20, 2022

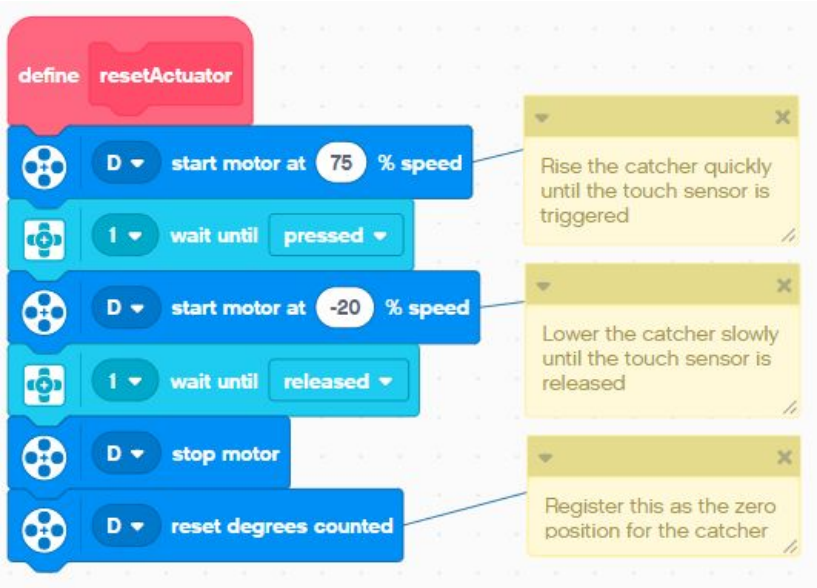
Lesson 3, Front Catcher Extension





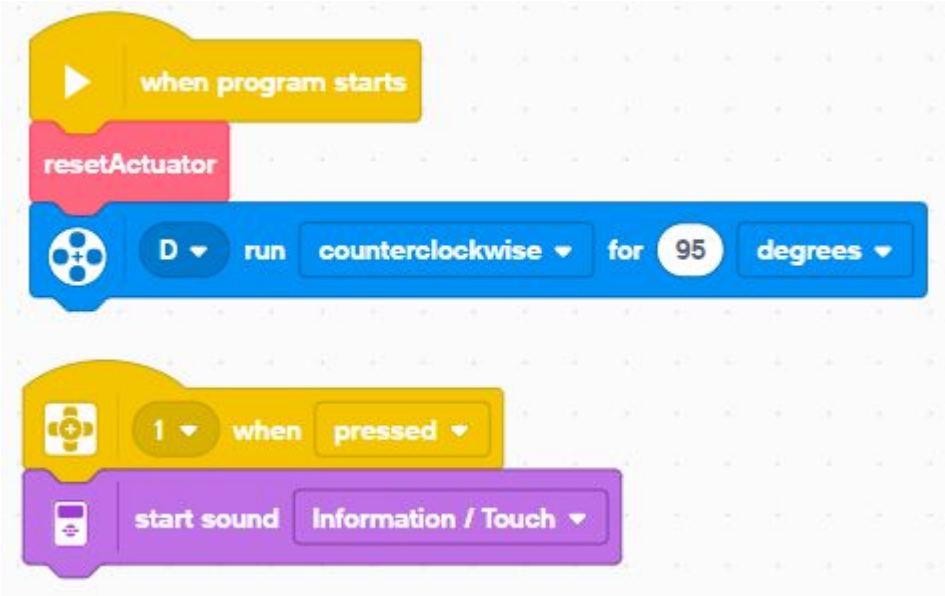
Lesson 3, Front Actuator Calibration

- Add a front catcher extension to trigger the touch sensor
- Create a MyBlock to reset the actuator connected to motor D



Lesson 3, Game Preparation

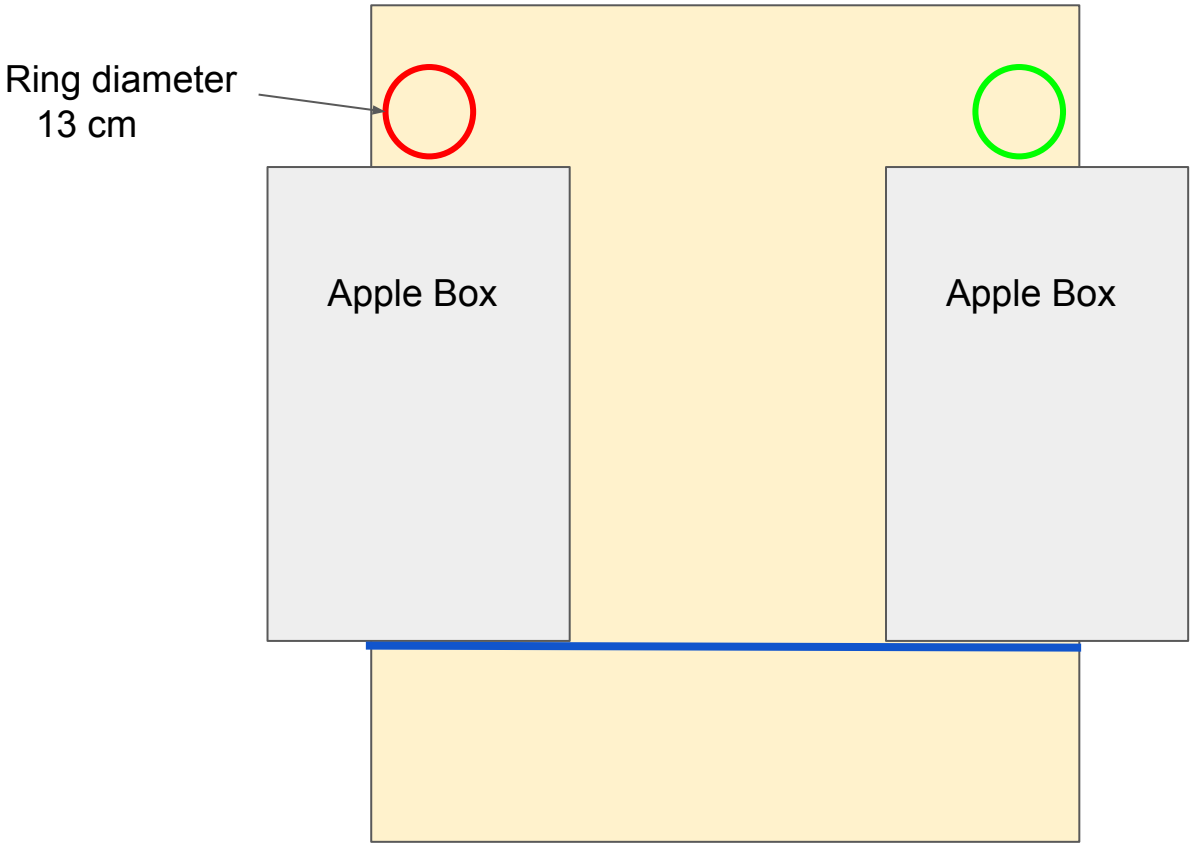
- In the game, the front catcher needs to capture 4 plastic rings on ground
- For that reason the catcher has to go as low as possible
  - If it goes too low, it impacts the robot movements
- Try angles 90, 95, and 100
- Use a ring to observe which angle gives enough force to keep the ring captured.
- For sound effect, play the Touch sound when the sensor is triggered.



Lesson 3, Ring Movement Game

- This is a timed competition
- Four rings have to be moved
  - For every missing rings, 5 seconds is added to the time
  - If all rings are missing, the run is disqualified
- The rings are originally behind a wooden block as marked in red
- The robot starts at base below the lower tape
- The rings has to be moved to behind another wooden block marked in green
- Finally, the robot has to return back base below the lower tape

Lesson 3, Apple Boxes on the table



The apple boxes are
50 cm deep
30 cm wide

The distance between the boxes is 32 cm

The table top is
75 cm wide

The center of the red rings is
8 cm from table top edge
8 cm from apple box

Lesson 4, Variables

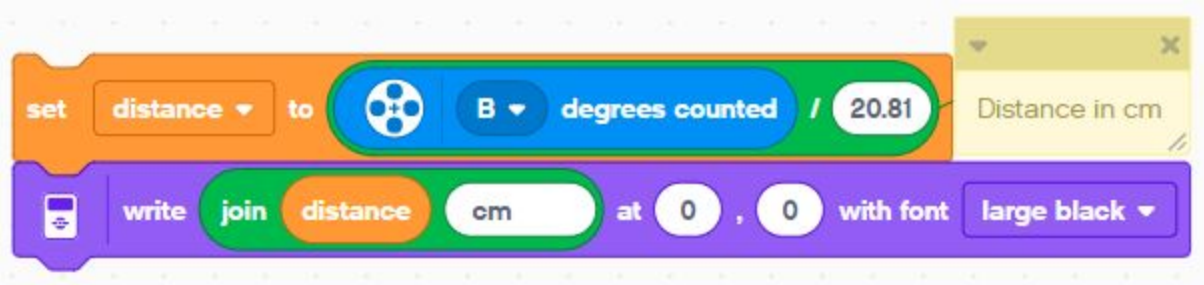
- To find a place where the robot can be parked the distance between the parked cars has to be measured and stored in a variable.
- Once the variable is large enough, the robot can start the parking maneuvers
  - We can start with the assumption that 25 cm is enough for the robot
- Create the variables by selecting
  - Variables: Make a Variable
- Init the minDistance value in the main program

Lesson 4, Parallel Parking Game

- This is a timed competition
- The robot starts below the lower tape
- When the robot moves, it uses the distance sensor to detect when a gap between the car starts
- If it sees the next car before the gap is large enough, it will try the next gap
- Once large enough gap is found
  - The robot could move forward a little bit (like the real cars)
  - The robot backs to right
  - The robot could move backward a little bit to get completely inside the found space
  - The robot backs to left to align with traffic
  - The robot shows the time used
- The distance from the passed car in cm is added to the time as seconds
  - Distance between the box and the rear bumper (not the cables)

Lesson 4, Hint

- The distance between the parked cars can be measured with the degrees counted in the B motor
- The degrees can be converted into cm by dividing the reading with 20.81
  - This is the constant used in the go block
- For debugging purposes, the distance can be shown in the EV3 display
  - The join operation is used to add the text cm after the distance value
  - The extra spaces after cm are used to clean the mess caused by different lengths in the value string



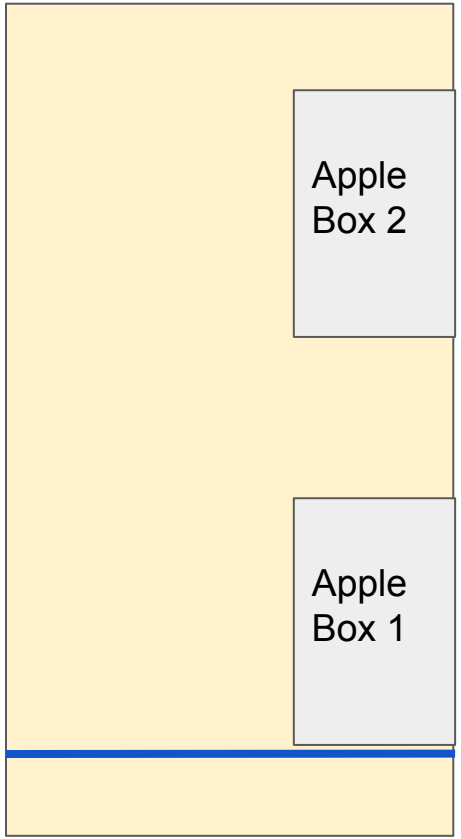
Lesson 4, Parked Cars

The apple boxes simulate the parked cars

Start with a distance of 30 cm between the apple boxes.
    The robot will park between the boxes

Reduce the distance to 20 cm.
    The robot will detect that there is not enough space.
    It will continue and pass the box 2.
    Even if there are no other boxes in front of box2
        The robot will park there once the free space
        Is confirmed

Note that the real distance is few cm longer than the target Distance due the fact that the degree counter of the motor is reset after the distance sensor has detected the free space.
The sound wave is a cone and thus impacted by the robot distance to an apple box.



Lesson 5, Color Sensor

- Color sensor can identify
  - The color underneath it
    - This is used to exit a loop
  - The amount of reflected light
    - Low value, such as 20 indicates a dark surface
    - High value, such as 80 indicates a light surface
    - This is used to follow an edge between a dark area and light area
- Color sensor calibration
  - Color identification doesn’t require any calibration
  - To calibrate the reflected light, the robot should be moved around the dark and light areas
    - The values need to be written down as they will be used in the program

Lesson 5, Control Theory

The **proportional control** is used in many cases during Amazing Maze season.

The basic idea is simple:

- If current observation is close to the target
  - minimal correction is required
- The further away the observation is from the given target
  - the stronger correction is required

The common practices are:

- Identify the current observation as process value (**PV**)
- Identify the target as set point (**SP**)
- Identify the correction as control value (**CV**)

Lesson 5, Proportional Control Formula

Normally the difference between PV and SP is know as Error

- We will use a more descriptive name **delta** for PV - SP

The control tuning parameter is the delta multiplier in CV calculation

- Normally this is known a kP
  - K identifies a constant
  - P identifies Proportional control
- We will start by using a more descriptive name **gain**

The formula is

CV = gain * (PV - SP) or gain * delta or kP * delta

Lesson 5, Racing around dohyo game

The color sensor is used to detect the reflected light at the dark border

- If the sensor reading is low, then the robot is too deep in the border
  - And can fall to the floor
- If the sensor reading is high, the the robot has left the border area
  - And be disqualified in the race
  - Or lead to a route causing the fall to the floor

The race start at the red line on the border

The race ends after one round around the dohyo

- When the robot detect the red line and records the used time

Lesson 5, Proportional Bias Control

In this use case, the robot should have a continuous bias towards right and use the proportional control only for small fine tuning.

CV = gain * (PV - SP) + bias

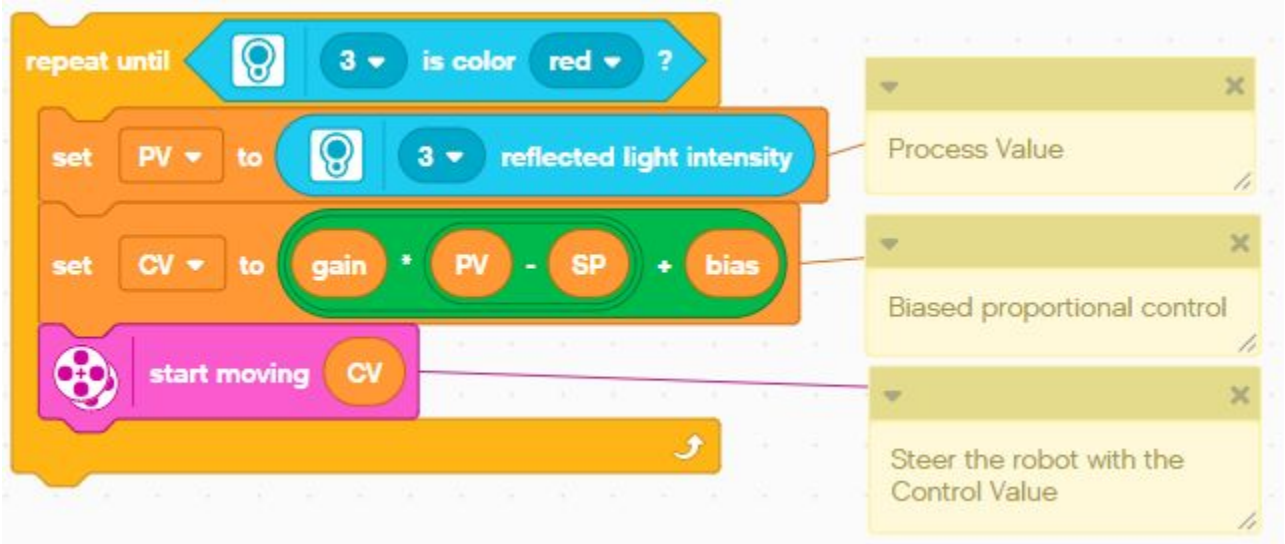
Recommended development steps

1. Set gain to 0 to find proper bias value
2. Set SP to around the middle value of the color sensor readings
3. The PV is the reflected light intensity from sensor 3
4. Start the gain adjustments from 0.1
    a. Observe how the positive values make things worse
    b. Try -0.1 and observe that it is not strong enough
    c. Try -1.0 and observe that it is too strong

Lesson 5, Code, Initialization

Create variables for bias, gain, and SP.
Set their values before the main loop start.
Try to run the robot at full speed.
If that is too fast reduce the speed to 80% or lower

Lesson 5, Code, Main Loop



Note how the color sensor is used to read both the color and the reflected light intensity.
The only parameter used in the start moving block is the steering direction

Lesson 5, Code, End the Race

Once the main loop is done due to the detection of the red tape

- The robot keeps moving and turning to right (25)
- It keeps moving as long it sees the red tape
- Once stopped, the robot shows the used time in seconds

Lesson 5, Challenge

The start moving block is internally using the **speed** control for motors B and C.

For faster times, the motor B could run at full 100 % **power** and the motor C power is then controlled with the CV value.