



Stack Rolls

EV3 for Game with Scratch Workshop

This file can be found under the **eAcademy**  **Workshops** page on the website

www.robofest.net

robofest@ltu.edu

248-204-3568

Room J233 Taubman Complex, LTU
21000 West 10 Mile Road, Southfield, MI 48075, USA

Please take Pre Assessment:

https://docs.google.com/forms/d/e/1FAIpQLScdoEHfiirs93sToeY3eZ6Ur9pA1DX86l8hev9v4LKXFHMDBA/viewform?usp=sf_link

This file can be found under the **eAcademy** on the website **www.robofest.net** along with online training and certifications



2021 Workshops

Presented by

Lawrence Technological
University
Computer Science

Course Overview

- 2021 Robofest competition Stack Rolls
 - Autonomous robot that get points by moving and stacking rolls
- SPbot introduction
- Using SPbot to solve the Stack Rolls challenge

2021 Robofest Competition

- Video overview
- Key tasks

Task 0: Finding the edge of the table

Task 1: Following the edge of the table

Task 2: Stop line following when you reach a corner

Task 3: Stop line following when you reach a given distance

Task 4: Finding a Paper Roll

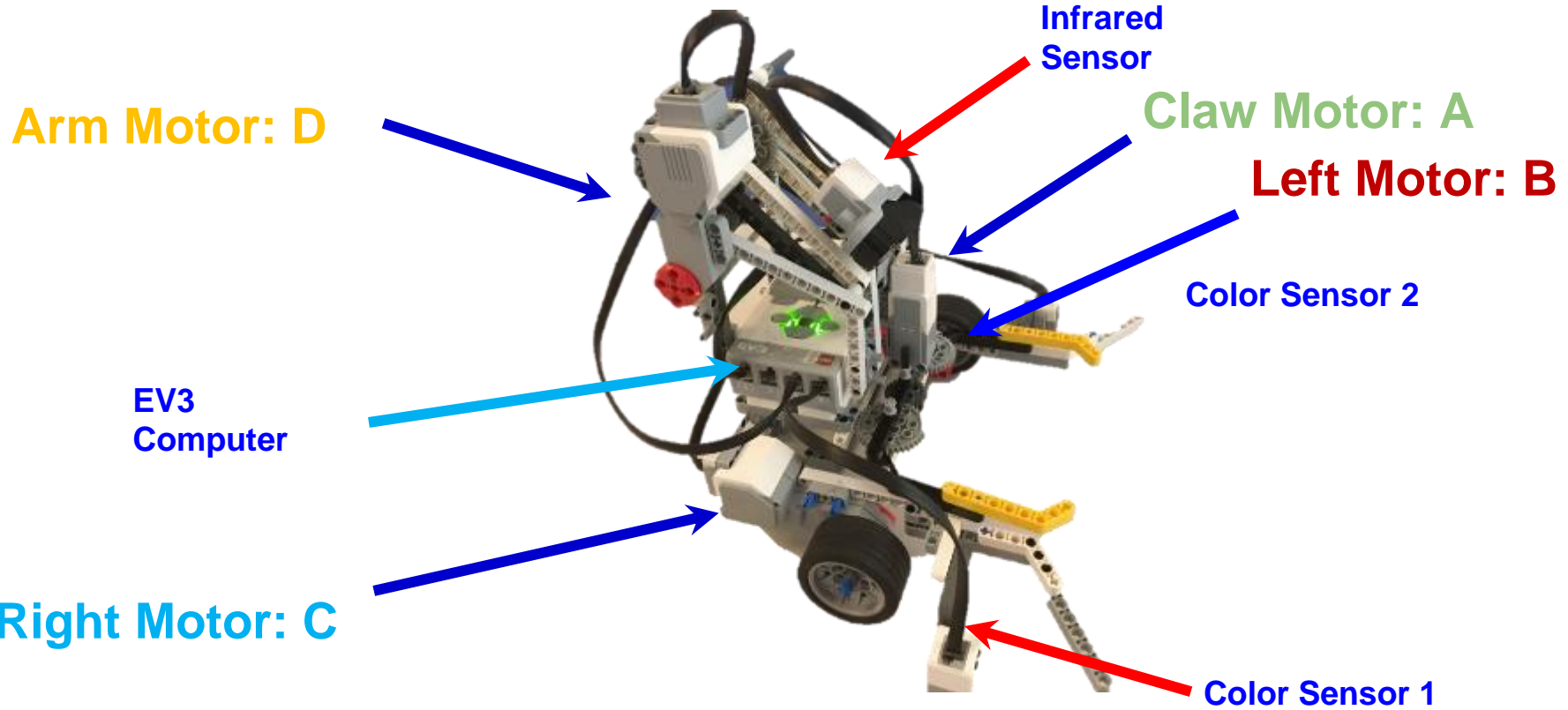
Task 5: Turning the robot

Task 6: Aligning the robot to an edge

Task 7: Manipulating Paper Rolls

Task 8: Building MyBlocks

LEGO EV3 robot used



Remember the connections!

- Left Motor connects to **B**
- Right Motor connects to **C**
 - If your motors are upside down forward will be backwards in your program
- Arm Motor connects to **D**
- Claw Motor connects to **A**
- Color sensor 1 (Right) connects to port no. **1**
- Color sensor 2 (Left) connects to port no. **2**
- Infrared sensor connects to port no. **3**

EV3 Versions Used

- Examples use EV3 Educational Version **1.2.2**
 - Download
 - <https://education.lego.com/en-us/downloads/mindstorms-ev3/software>
- EV3 Firmware version: **V1.10E**
- Presentation and all example programs are available at robofest.net under Tech Resources

Brick Overview

Wireless Connection Status icons (from the left)



Bluetooth enabled but not connected or visible to other Bluetooth devices



Bluetooth enabled and visible to other Bluetooth devices



Bluetooth enabled and your EV3 Brick is connected to another Bluetooth device



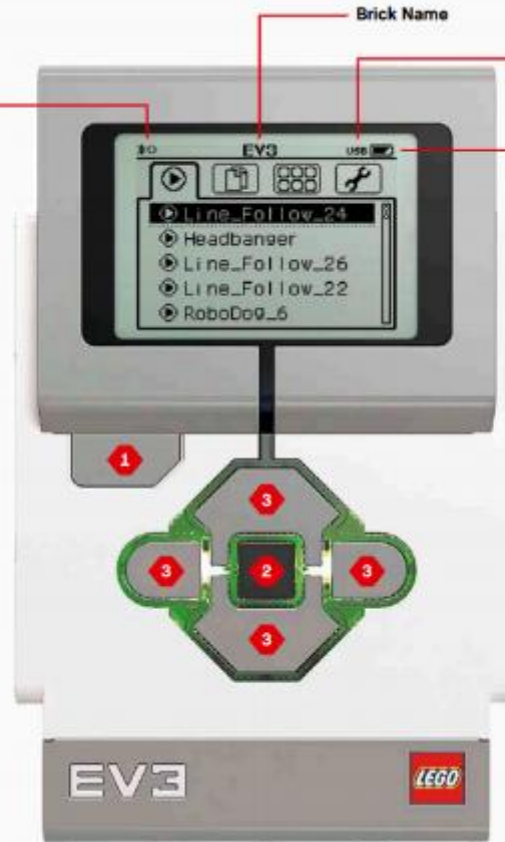
Bluetooth enabled and visible and your EV3 Brick is connected to another Bluetooth device



Wi-Fi enabled but not connected to a network



Wi-Fi enabled and connected to a network



USB

USB connection established to another device



Battery level

Brick Buttons

1. Back

This button is used to reverse actions, to abort a running program, and to shut down the EV3 Brick.

2. Center

Pressing the Center button says "OK" to various questions—to shut down, to select desired settings, or to select blocks in the Brick Program App. You would, for example, press this button to select a checkbox.

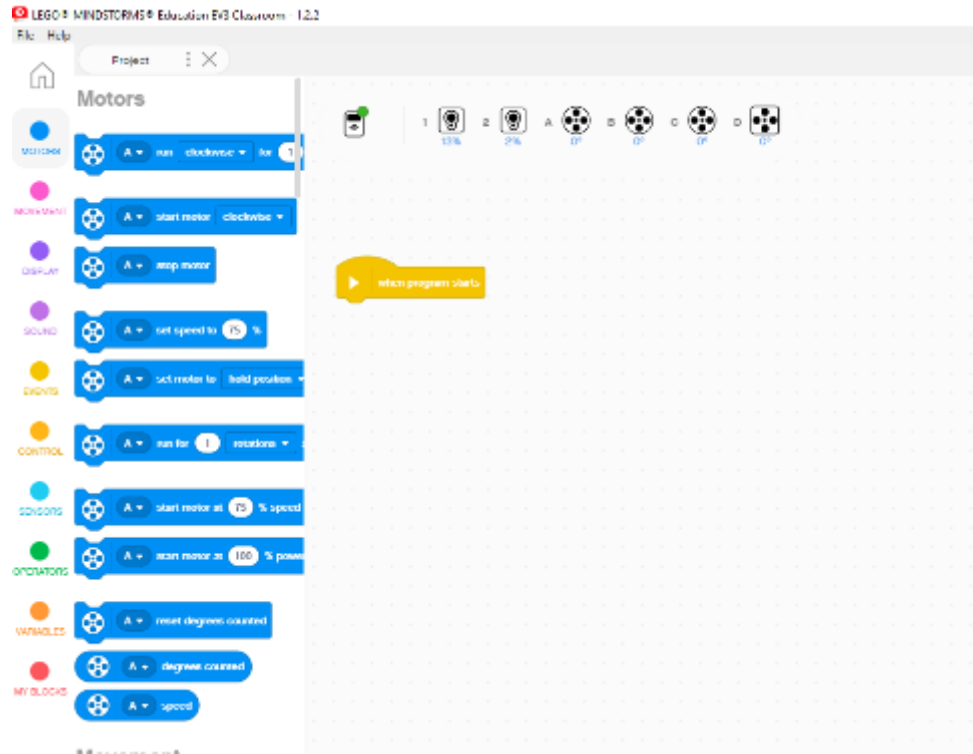
3. Left, Right, Up, Down

These four buttons are used to navigate through the contents of the EV3 Brick.

Connect to Brick

- Two options
 - USB Cable
 - Bluetooth

Scratch Programming Environment

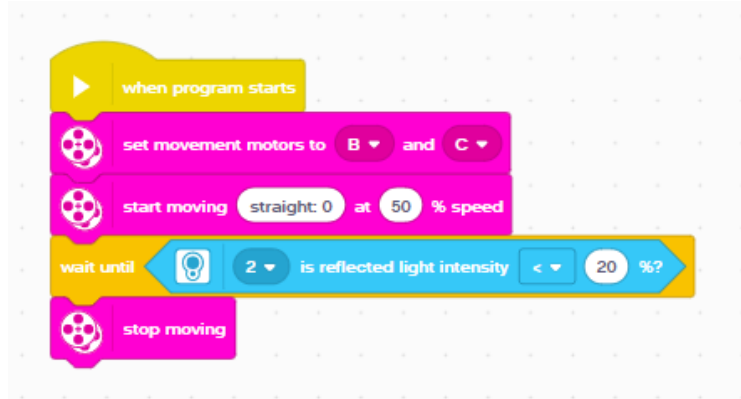


Task 0

Finding the edge of the table

Task 0: Example Solutions

- Using a wait block
- Using a loop block



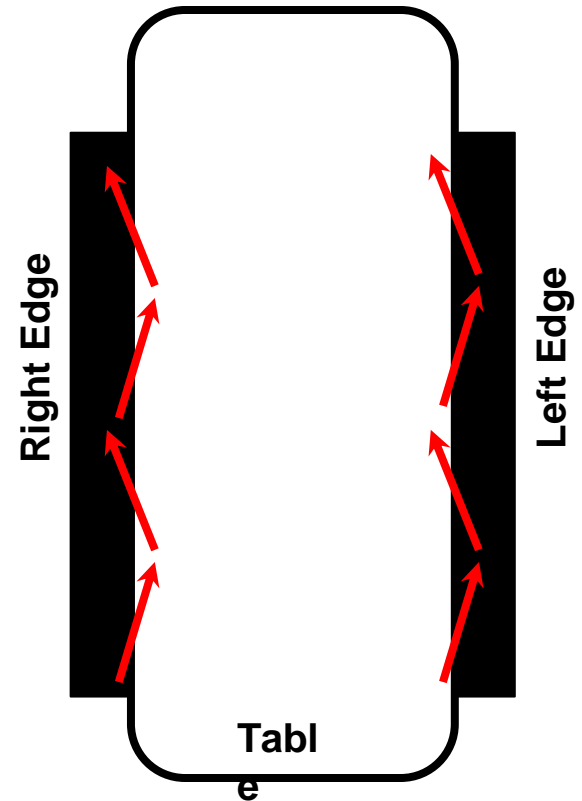
YouTube: <https://youtu.be/NgXrOGvQsII>

Task 1

Following the edge of the table

Following The Edge Of The Table

- Use the zig-zag method to follow the edge of the table
- Edge following is also referred to as line following
- The zig-zag method requires the use of a sensor determine when the robot is on or off the table



Following The Edge Of The Table

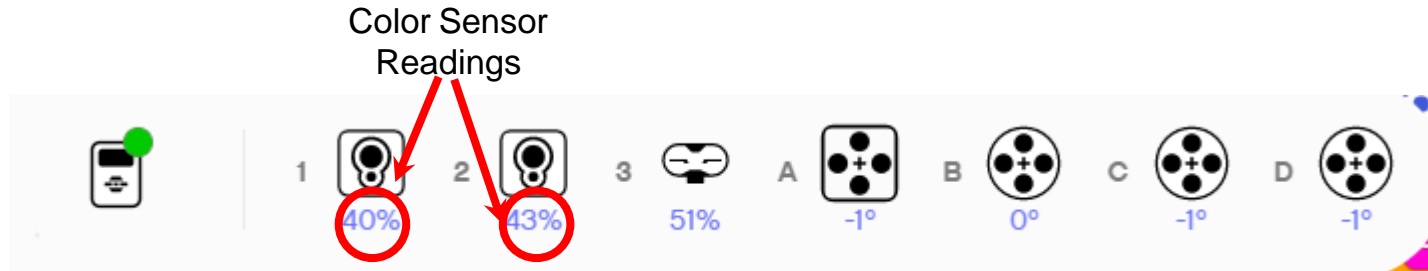
- Get color sensor values to determine when the robot is **on** or **off** the table. We will use the color sensor in Reflective Light Intensity mode.

- Color Sensor 1

- On table = _____ (60)
- Off table = _____ (20)

- Color Sensor 2

- On table = _____ (60)
- Off table = _____ (10)



Following The Edge Of The Table

- For light sensor #2 settings example
 - On table = 40
 - Off table = 0
 - Median threshold = $(40+0)/2 = 20$
- Two cases
 - Light sensor reading > 20 . On table.
 - Light sensor reading < 20 . Off table.

Simple Line Following Algorithm

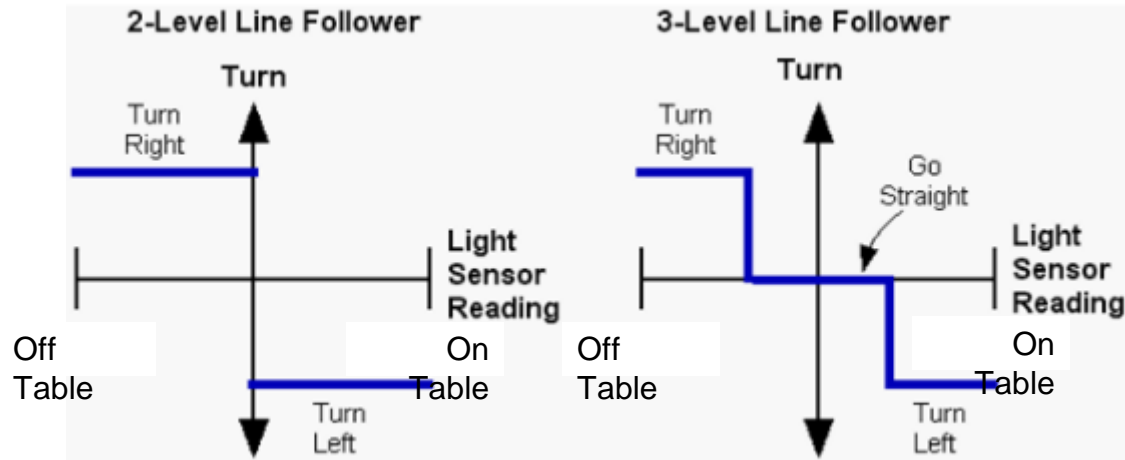
```
when program starts
  set movement motors to B and C
  forever loop
    if light sensor 2 is reflected light intensity < 20% then
      start moving right: 15 at 50% speed
    else
      start moving left: -15 at 50% speed
```

The image shows a Scratch-style block diagram for a line following algorithm. It starts with a yellow 'when program starts' block. This is followed by a pink 'set movement motors to B and C' block. A yellow 'forever' loop block contains an 'if' block. The 'if' block has a light sensor icon, the number '2', the text 'is reflected light intensity', a less-than sign '<', the number '20', and a '%' symbol. The 'then' part of the 'if' block contains a pink 'start moving right: 15 at 50% speed' block. The 'else' part of the 'if' block contains a pink 'start moving left: -15 at 50% speed' block. The 'forever' loop block has a small arrow icon at the bottom right.

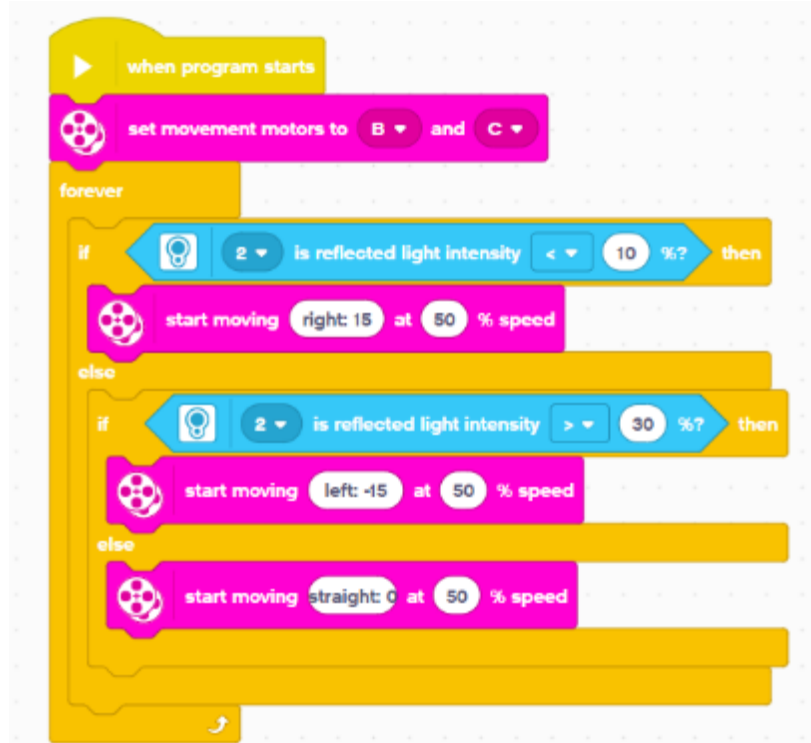
YouTube: <https://youtu.be/EfBdoJESlro>

How to improve our line following algorithm

- The zig-zag method can cause a bumpy response
- To improve the response, you can use a 3-level line follower (concept shown below)



How to improve our line following algorithm



```
when program starts
  set movement motors to B and C
  forever
    if 2 is reflected light intensity < 10 %? then
      start moving right: 15 at 50 % speed
    else
      if 2 is reflected light intensity > 30 %? then
        start moving left: -15 at 50 % speed
      else
        start moving straight: 0 at 50 % speed
```

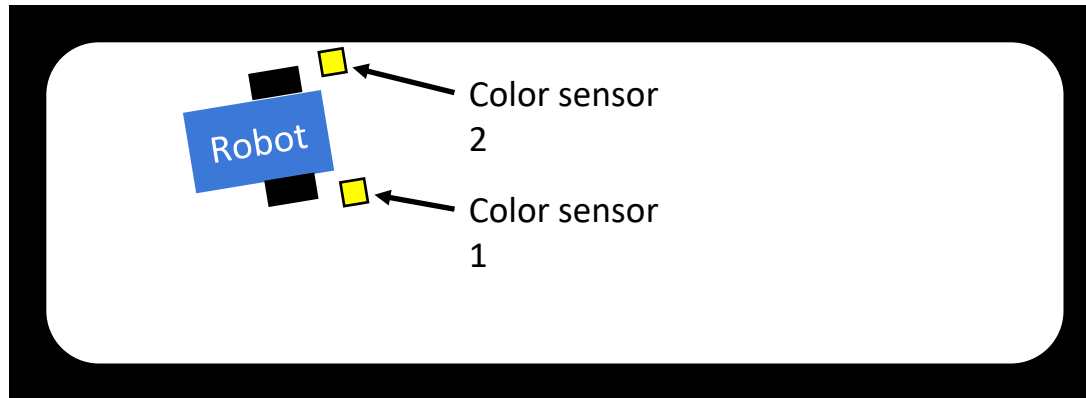
The image shows a Scratch script for a line following robot. It begins with a 'when program starts' block, followed by a 'set movement motors to B and C' block. A 'forever' loop contains three conditional blocks. The first is an 'if' block with a light sensor icon, '2' in a dropdown, 'is reflected light intensity', '<' in a dropdown, '10' in a text field, and '%?' in a dropdown. The 'then' block contains a 'start moving' block with 'right: 15' in a dropdown, 'at 50' in a text field, and '% speed' in a dropdown. The 'else' block contains another 'if' block with a light sensor icon, '2' in a dropdown, 'is reflected light intensity', '>' in a dropdown, '30' in a text field, and '%?' in a dropdown. Its 'then' block contains a 'start moving' block with 'left: -15' in a dropdown, 'at 50' in a text field, and '% speed' in a dropdown. The 'else' block contains a 'start moving' block with 'straight: 0' in a dropdown, 'at 50' in a text field, and '% speed' in a dropdown.

Task 2

Line following to the corner of the table

Line following to the corner

- One method of line following to the corner is to follow the edge of the table with one color sensor and detect the end of the table with a other color sensor
 - Sensor 1 used to locate the end of the table
 - Sensor 2 used to follow the edge of the table

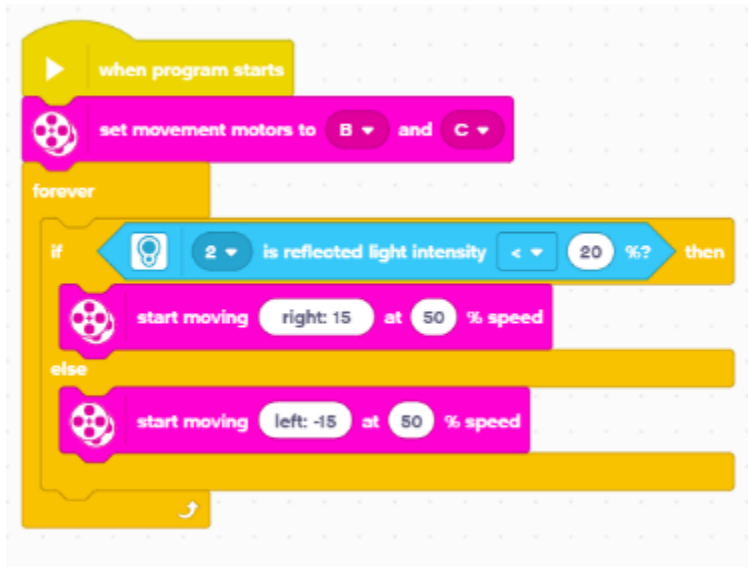


Line following to the corner

- Couple comments regarding moving around the table
 - It is possible to travel around the edge of the table with only one color sensor, but it is more difficult and potentially less reliable than using two colors sensors
 - Black tape is used to denote zones. We can use the black tape to line follow to the end of a **zone**.

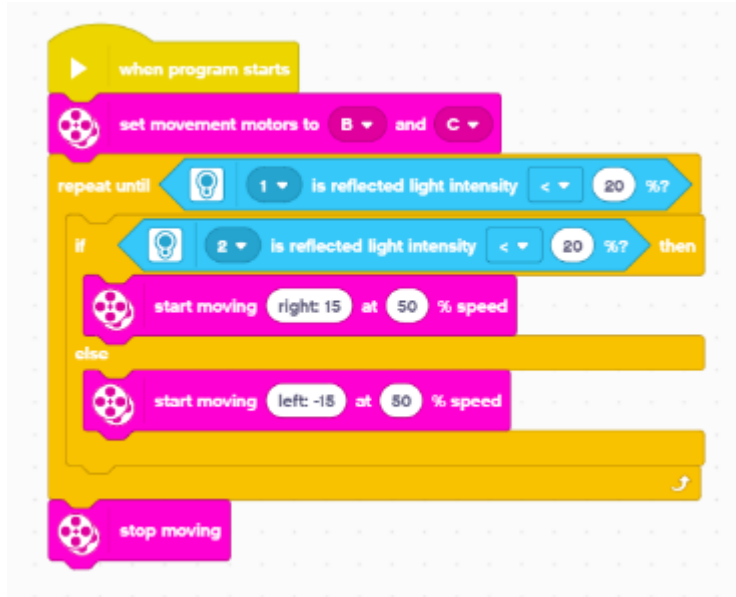
Line following to the corner

- Recall our line following program
- Let's modify the program to stop when the robot reaches the end of the table



Using this program, the robot will line follow continuously. How can we make the robot stop when it reaches a corner?

Line following to the corner



YouTube: <https://youtu.be/oh8a5bk0PkY>

Task 3

Line following a given distance

Line following a given distance

- Approach

- Modify LineFollow program to stop when the robot travels a given distance

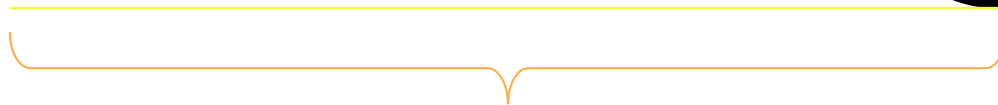
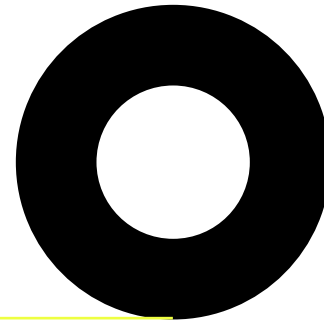
- Tools needed

- Line following
- Measure distance traveled

Measuring Distances

- How do we measure distance traveled?
- Let's determine how far the robot travels moving forward for 2 seconds

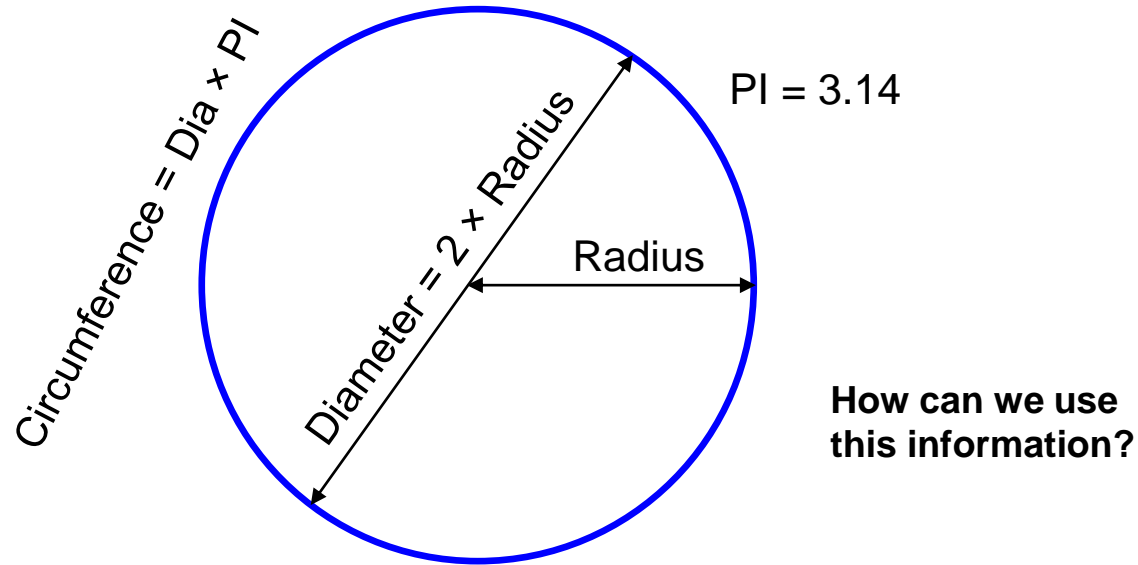
**Compute distance traveled
by measuring the number
of rotations of the wheel**



Distance

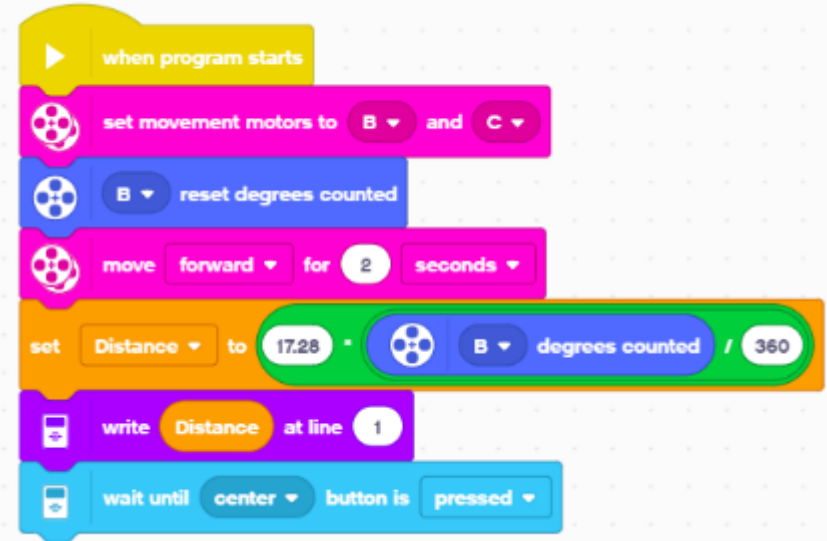
Measuring Distances

- Use the wheel geometry



Measuring Distances

- For each rotation of the wheel, the robot travels (Wheel Diameter) x (PI)
 - Distance = (Wheel Diameter) x (PI) x (# Rotations)
 - Distance = (5.5 cm) x (PI) x (# Rotations)
 - Distance = (17.28 cm) x (# degrees/360)



YouTube: <https://youtu.be/r0psO1aK0ME>

Measuring Distances

● Example

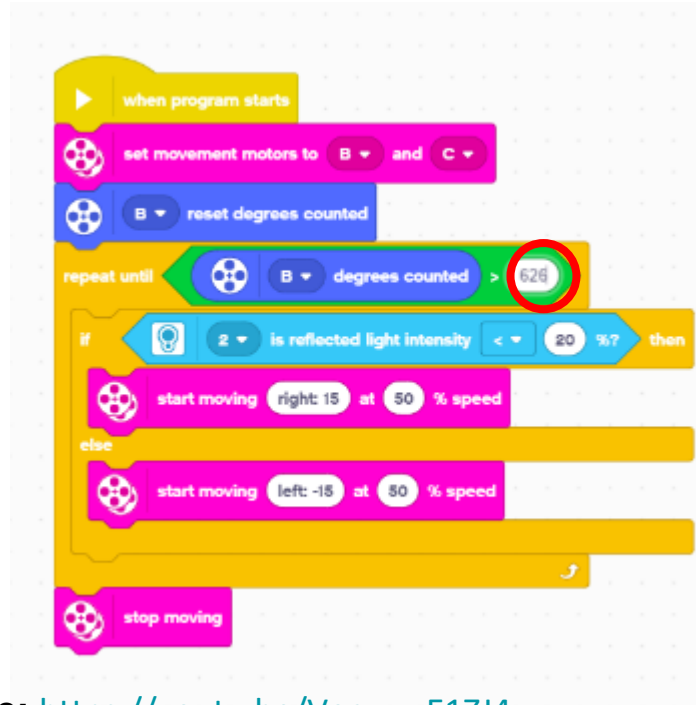
- Let's program the robot to line follow for 30 cm
 - Distance = 30 cm
- Number of rotations
 - Distance = (Wheel Diameter) x (PI) x (# Rotations)
 - Distance = (Wheel Diameter) x (PI) x (# degrees/360)
 - Solve for (# degrees)

$$(\# \text{ degrees}) = \frac{\text{Distance} \times 360}{(\text{Wheel Diameter}) \times (\text{PI})}$$

$$(\# \text{ degrees}) = \frac{30 \text{ cm} \times 360}{(5.5 \text{ cm}) \times (\text{PI})} = 626 \text{ degrees}$$

Line following a given distance

- Line follow a desired distance



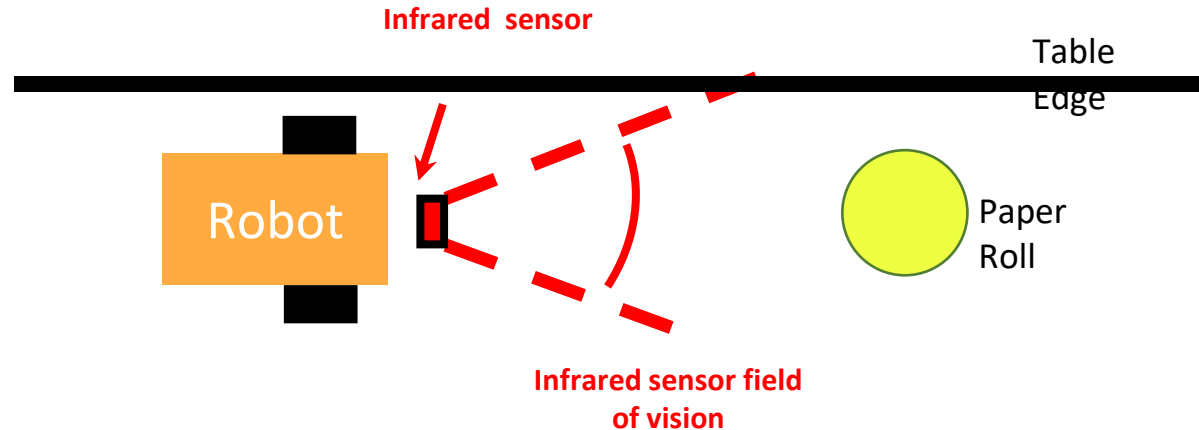
YouTube: https://youtu.be/Vocwy_E1ZJ4

Task 4

Finding a Roll

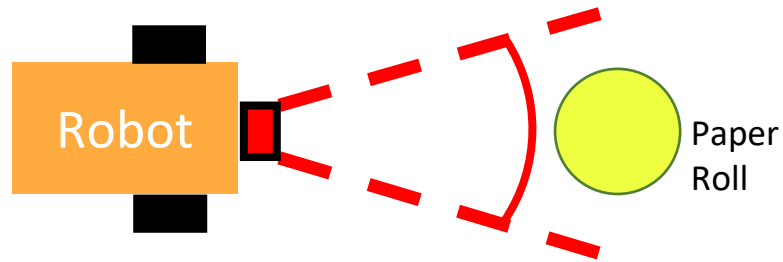
Finding a Rolls

- We can use the infrared sensor to determine if an object is near the robot
- Here we will assume that we are following the edge of the table and wish to stop the robot once a paper roll close to the robot



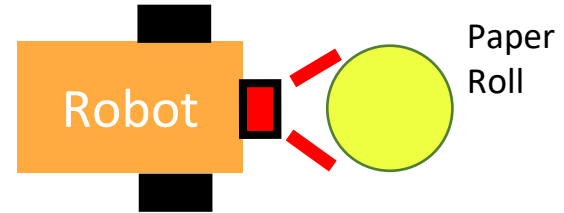
Finding a Paper Roll

- Here we will use our line following program to follow the edge of the table and stop the robot when the Paper Roll is close to the infrared sensor



Infrared sensor will read high values when the Paper Roll is far away

Starting Position

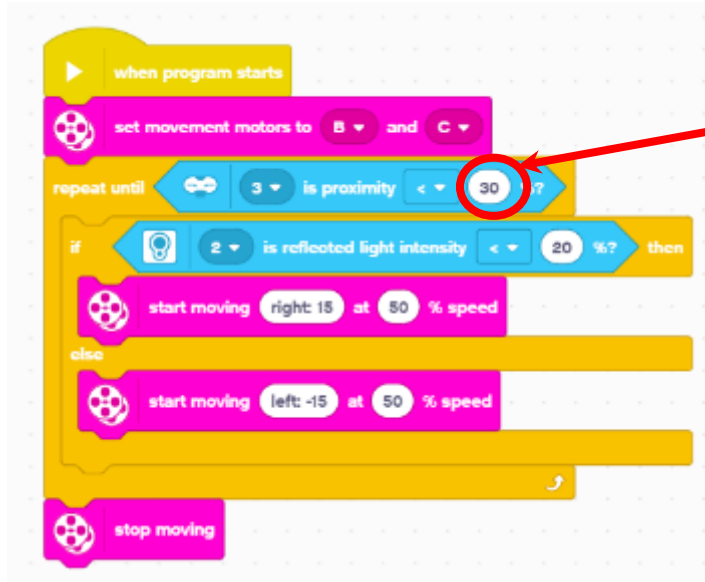


Infrared sensor will read low values when the Paper Roll is close to the robot

Final Position

Finding a Paper Roll

- Now, we travel along the edge of the table and stop if we find a Paper Roll



We chose a low threshold. We can determine the appropriate value by testing the sensor readings with a Paper Roll near the front of the robot.

YouTube: <https://youtu.be/vy89KXPufcQ>

Task 5

Turning the robot

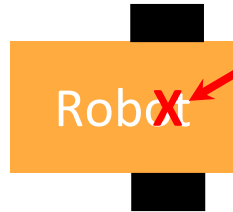
Turning The Robot

- For our example here, we wish to turn the robot 90 degrees
- There are several methods for turning a tripod robot. We will focus on two methods
 - “Spin” turn
 - “Swing” turn

90 Degree Spin

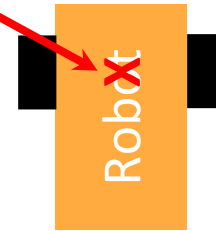
- Let's have the robot spin 90 degrees CCW
- The robot will rotate about center of the drive wheels

Starting Position



Center of
drive
wheels

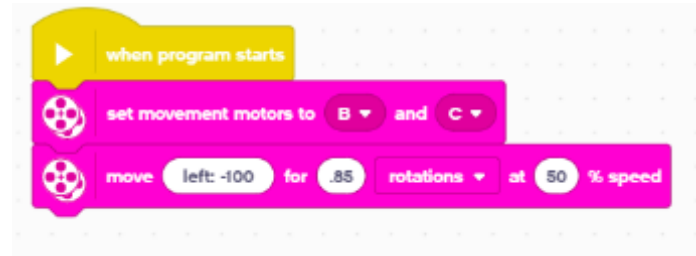
Final Position



90 Degree Spin

- To spin 90 degrees CCW, we use the Move block as shown here

- Set the steering to -100. This causes:
 - Right wheel to rotate forward
 - Left wheel to rotate reward
 - Equal and opposite rotations



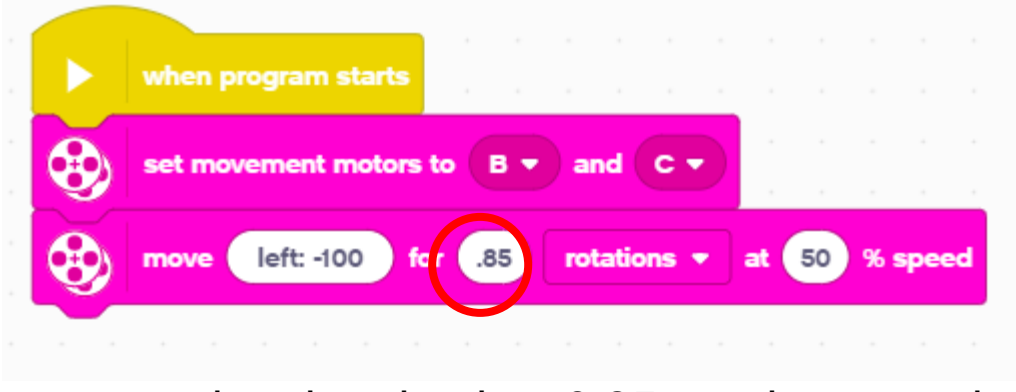
- Now, we need to determine the correct number of rotations

90 Degree Spin

- You can determine the proper number of rotations mathematically; however, the result typically needs some adjustment due to lash in the motors
- For today's class, we will use trial and error to find the number of rotations that cause the robot to turn 90 degrees

90 Degree Spin

- We can use one block to spin the robot



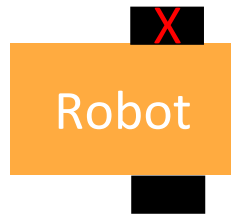
- For our sample robot, it takes 0.85 rotations to spin the robot 90 degrees

YouTube: https://youtu.be/Wdej_zMUEag

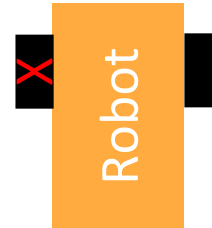
90 Degree Swing

- Let's have the robot swing 90 degrees CCW
- The robot will rotate about a locked wheel (denoted by red X)

Starting Position

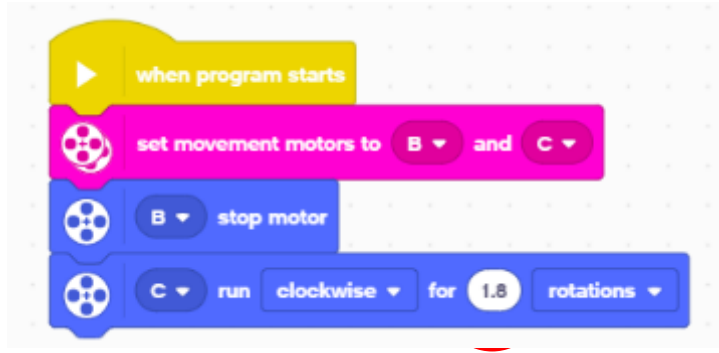


Final Position



90 Degree Swing

- To swing, we lock the left motor and power the right motor to turn the robot



- For our sample robot, it takes 1.8 rotations to swing the robot 90 degrees

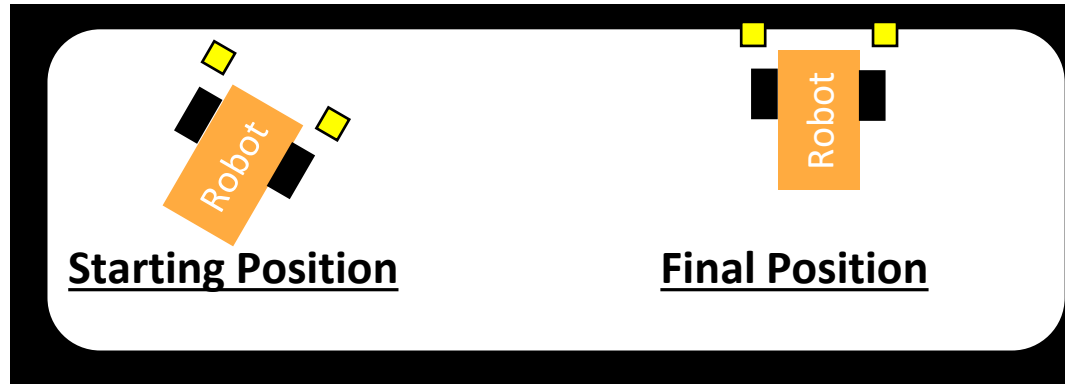
YouTube: <https://youtu.be/PBcYS8ugfRM>

Task 6

Aligning the robot to an edge

Aligning the robot to an edge

- In some situations we desire align with robot to an edge of the table as shown below
- Assuming the starting position below, how can we program the robot to reach the final position that is aligned with the edge of the table?



Aligning the robot to an edge

- Travel until color sensor #2 reaches the edge, swing robot until it is aligned with the edge



YouTube: <https://youtu.be/TjrYUN0y6Dc>

Align if either side detects edge first

The diagram shows a sequence of programming blocks for a robot's alignment logic:

- when program starts** (yellow block)
- set movement motors to D and C** (pink block)
- start moving straight: 0 at 20 % speed** (pink block)
- wait until** (green block) with two conditions: **2 is reflected light intensity < 20 %?** or **1 is reflected light intensity < 20 %?**
- stop moving** (pink block)
- if** (yellow block) with condition **2 is reflected light intensity < 20 %?** then:
 - C start motor at 20 % speed** (blue block)
 - D stop motor** (blue block)
 - wait until** (green block) with condition **1 is reflected light intensity < 20 %?**
 - stop moving** (pink block)
- else** (yellow block):
 - C start motor at 20 % speed** (blue block)
 - C stop motor** (blue block)
 - wait until** (green block) with condition **2 is reflected light intensity < 20 %?**
 - stop moving** (pink block)

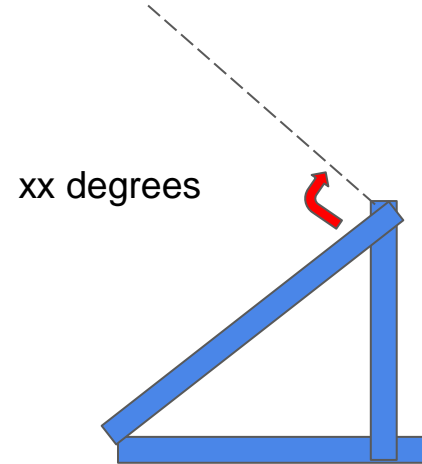
Task 7

Manipulating Paper Rolls
Moving the Arm and Claw

How to Control the Arm and Claw

- Time
- Rotation(encoder degrees)

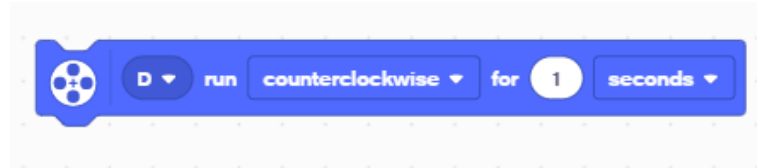
Note: To use rotation, you need to determine what direction to run the motor to raise the arm. For this workshop, the arm is run counter clockwise (negative direction)



Arm all the way down
0 degrees

Moving the Arm- Using time

- Advantages
 - Simple
 - Easy to program
 - Will not get stuck
- Disadvantages
 - Can be imprecise
 - Repeatability
- How to do it
 - Set a motor block to “seconds”
 - Select the motor port
 - Select the direction
 - Select the duration



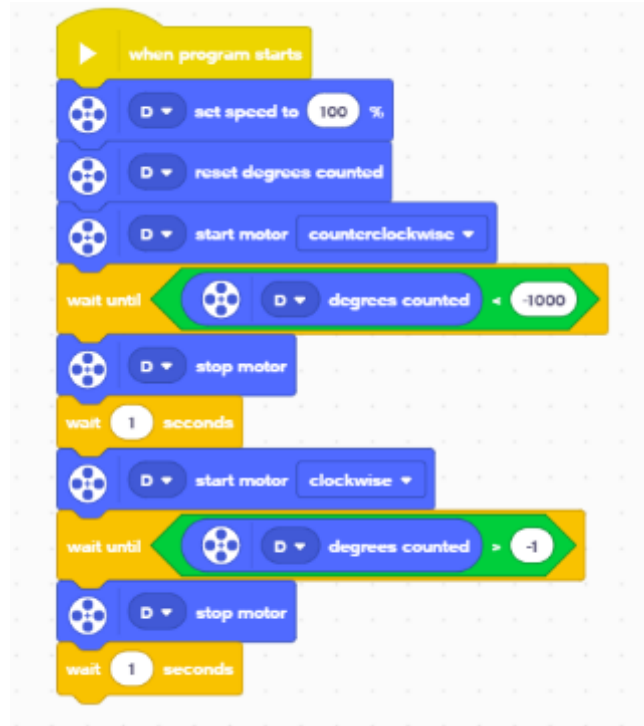
Raise
Arm



Lower
Arm

Moving the Arm- Using encoder

- Advantages
 - More precise
 - More repeatable
- Disadvantages
 - More difficult to program
 - Can get stuck
- How to do it
 - Establish a “zero” point
 - Determine direction of motor
 - Set limits



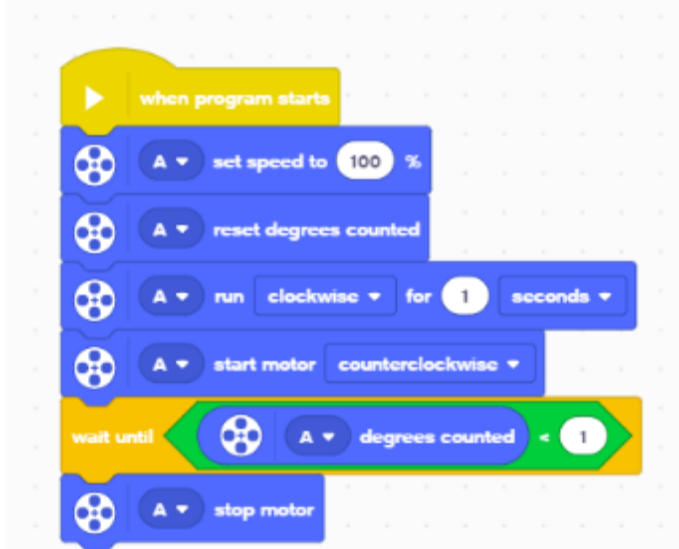
Raise arm,

Wait 1 sec

Lower arm back to 0

Moving the Claw

- May need to overdrive motor to get enough grip
- Use encoder to open back to zero position

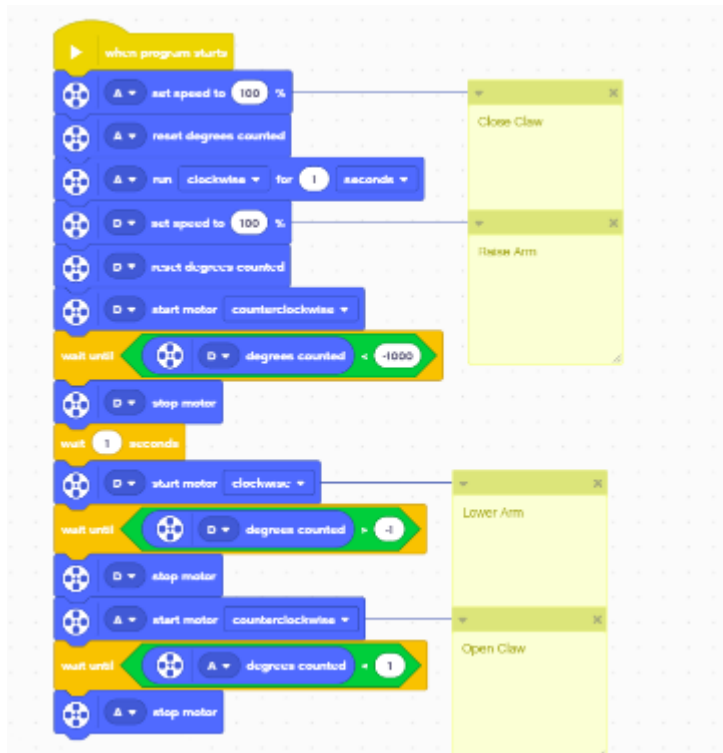


Set Power and zero position at beginning of program

Use time to close

Use rotation to open

Combine Arm and Claw Movement



YouTube: <https://youtu.be/C3rSA4EsmDM>

Task 8

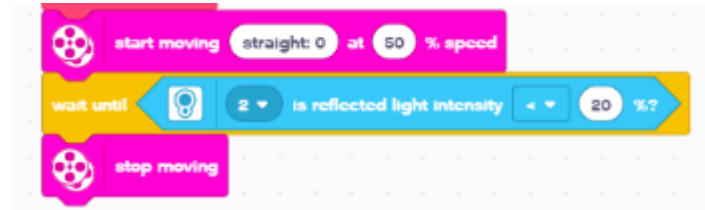
Building MyBlocks

My Blocks

- Solving the Robofest Game challenge will typically require a fairly large EV3 program (around 100 blocks is not unreasonable)
- Very large programs can be difficult to understand, navigate and use
- To alleviate this issue, the EV3 software has a My Block Builder to create custom blocks that can replace sections of your program

My Blocks

- For example, let's assume you have a section of code that completes the following:
 - Move forward until the edge of the table is found with color sensor 1, then stop
- The code may look like this

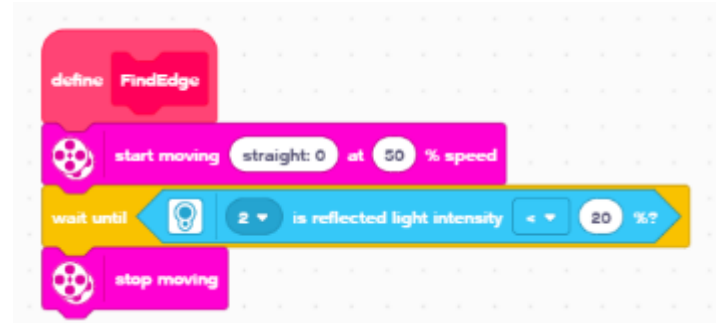
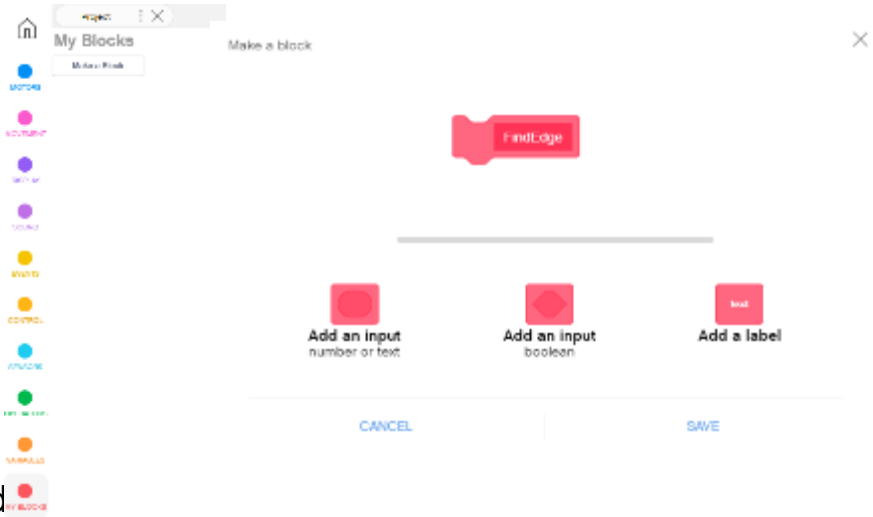


- My blocks will allow us to convert this to a single block

My Blocks

- Creating a My Block
 1. Click on “MyBlocks”
 2. Click on “Make a Block”
 3. Create a name “FindEdge” for the block
 4. Click “save
 5. Add blocks to be run when My Block is used

This creates a My Block called FindEdge that will be located in the My Blocks Pallet



Putting It All Together

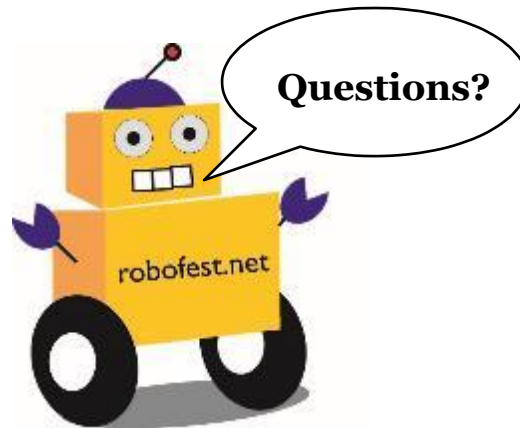
- In this course we learned about
 - Finding the edge of the table
 - Following the edge of the table
 - Stop line following
 - When you reach a corner
 - When you reach a given distance
 - Finding a Paper Roll
 - Turning the robot
 - Aligning the robot to an edge
 - Manipulating Paper Rolls
 - Building MyBlocks

Test for Knowledge

Post Assessment Link:

[https://docs.google.com/forms/d/e/1FAIpQLSe1jcaCRqfyroYfyv1F-2HAe2Y8BsBNdlWwHfPv59IKaoGOcQ/viewform?usp=sf link](https://docs.google.com/forms/d/e/1FAIpQLSe1jcaCRqfyroYfyv1F-2HAe2Y8BsBNdlWwHfPv59IKaoGOcQ/viewform?usp=sf_link)

Little Robots, Big Missions



robofest@LTU.edu
LTU Computer Science



2021 Workshops

Sponsored by

Lawrence Technological
University
Computer Science