ROBOFEST 2021 GAME

Stack Rolls

RobotC Vex IQ 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

LAWRENCE TECHNOLOGICAL UNIVERSITY
ROBOFEST

2021 Workshops

Sponsored by

Lawrence Technological University

Computer Science

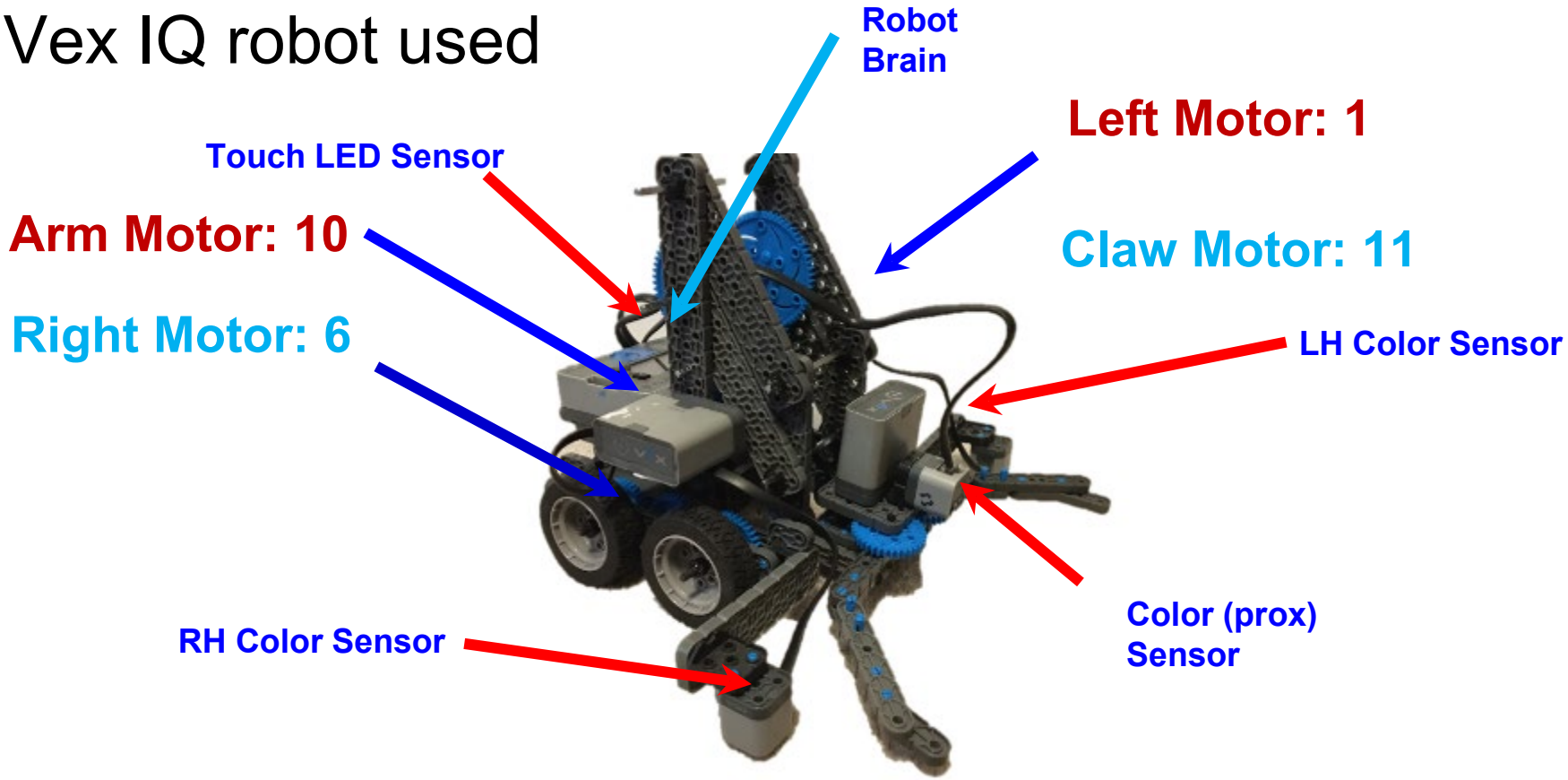
Course Overview

- 2021 Robofest competition Stack Rolls
  - Autonomous robot that get points by moving and stacking rolls
- Workshop robot introduction
- Using the workshop robot 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 Functions

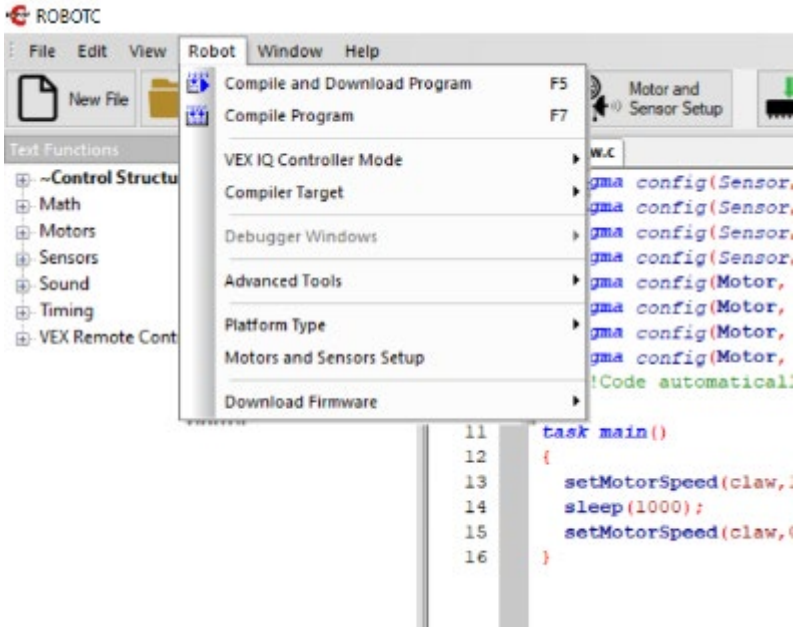
Vex IQ robot used



Remember the connections!

- Left Motor connects to **1**
- Right Motor connects to **6**
- LH Color sensor connects to port no. **7**
- RH Color sensor connects to port no. **5**
- Touch LED sensor connects to port no. **4**
- Color(prox) sensor connects to port no. **2**
- Arm Motor connects to **10**
- Claw Motor connects to **11**

Setting Up The ROBOTC Environment



Under Robot Menu

Vex IQ Controller Mode- Autonomous

Platform Type- Vex IQ

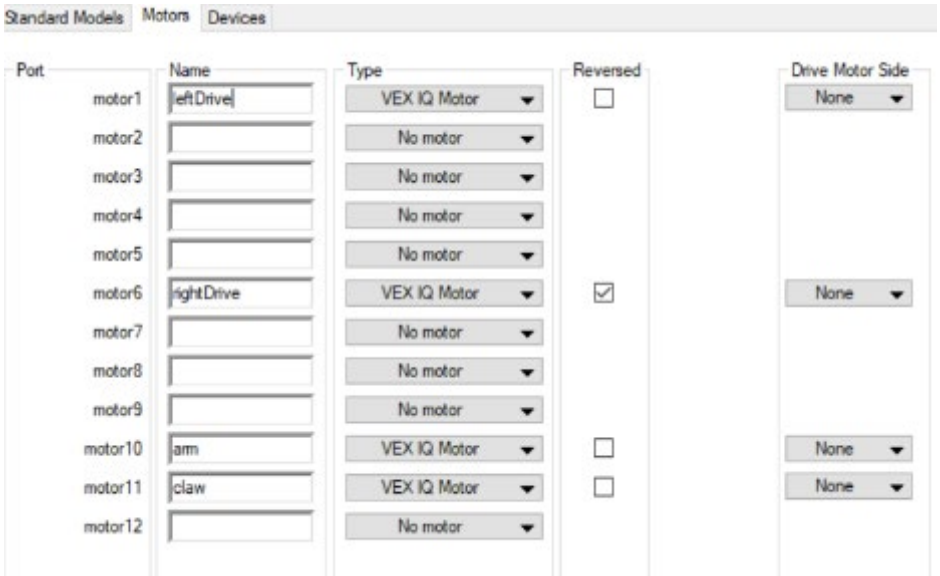
Uncheck Natural Language

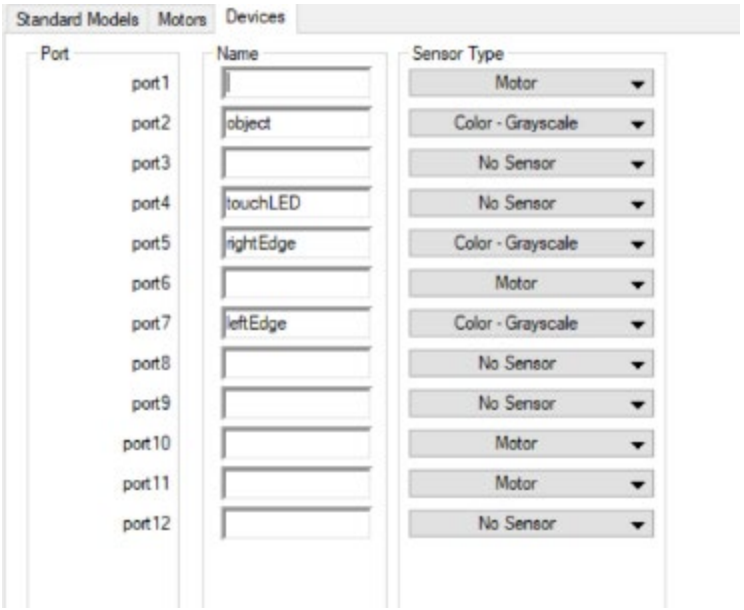
Motors and Sensor Setup Reviewed on the next slide

Setting Up The ROBOTC Environment

- The first time you use a Vex IQ robot with ROBOTC, you need to connect and download the ROBOTC firmware

Setting Up Motors and Sensors





Setting Up Motors and Sensors

- Once the motors and sensors at set up, ROBOTC will generate code to configure them
- We will use this code as a template for all programs we write in this course

```
1  #pragma config(Sensor, port2,  object,         sensorVexIQ_ColorGrayscale)
2  #pragma config(Sensor, port4,  touchLED,       sensorNone)
3  #pragma config(Sensor, port5,  rightEdge,      sensorVexIQ_ColorGrayscale)
4  #pragma config(Sensor, port7,  leftEdge,       sensorVexIQ_ColorGrayscale)
5  #pragma config(Motor,  motor1,          leftDrive,     tmotorVexIQ, PIDControl, reversed, encoder)
6  #pragma config(Motor,  motor6,          rightDrive,    tmotorVexIQ, PIDControl, encoder)
7  #pragma config(Motor,  motor10,         arm,           tmotorVexIQ, PIDControl, encoder)
8  #pragma config(Motor,  motor11,         claw,          tmotorVexIQ, PIDControl, encoder)
9  //*!!Code automatically generated by 'ROBOTC' configuration wizard               !!*//
10
```

Task 0

Finding the edge of the table

Task 0: Example Solution

```
task main()
{
  wait1Msec(1000);
  setMotorSpeed(leftDrive,50);//start robot
  setMotorSpeed(rightDrive,50);
  while(getColorValue(leftEdge)>100){//wait for dark
  }
  setMotorSpeed(leftDrive,0);// stop robot
  setMotorSpeed(rightDrive,0);
}
```

Program: FindEdge.c

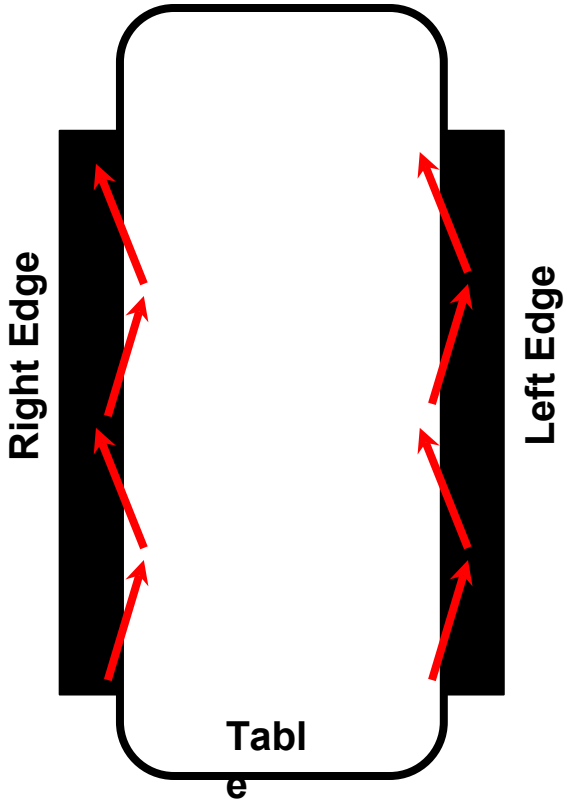
YouTube:
https://youtu.be/Q-2CG9p8fR0

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



Reading Sensors Values

- We can write a program to display the sensor values on the IQ LCD screen as well

Following The Edge Of The Table

- For light sensor #2 settings example
  - On table = 60
  - Off table = 10
  - Median threshold = (60+10)/2 = 35

- Two cases
  - Light sensor reading > 35. On table.
  - Light sensor reading < 35. Off table.

Simple Line Following Algorithm

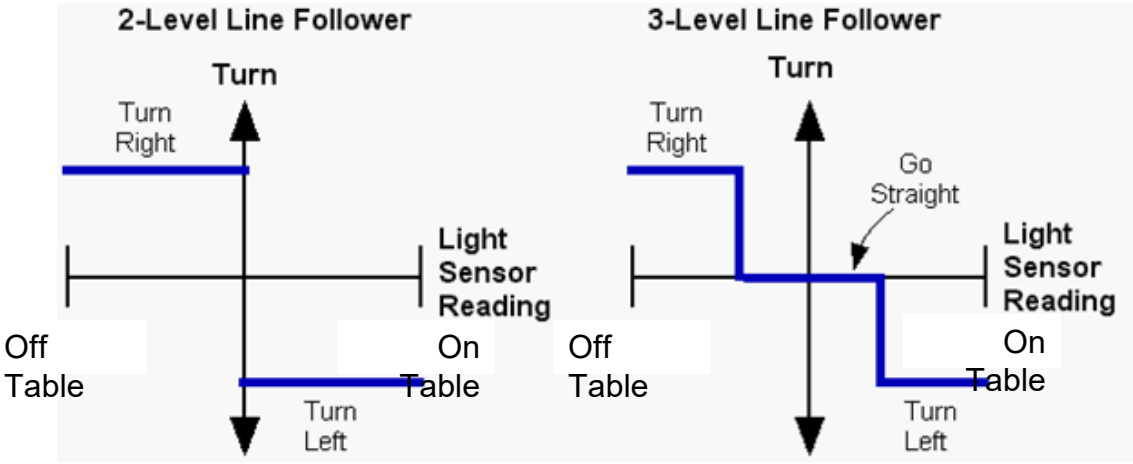
```
task main()
{
  while(true){
  if(getColorValue(leftEdge)<100){//if dark
    setMotorSpeed(leftDrive,50);
    setMotorSpeed(rightDrive,35);
  }
  else{
    setMotorSpeed(leftDrive,35);
    setMotorSpeed(rightDrive,50);
  }
  }
}
```


Program: LineFollow.c

YouTube:
https://youtu.be/AFRbJX_IpcE

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)



Improved line following algorithm

```
task main()
{
  while(true){
  if(getColorValue(leftEdge)>150){//if light
    setMotorSpeed(leftDrive,35);
    setMotorSpeed(rightDrive,50);
  }
  else if(getColorValue(leftEdge)<50){//if dark
    setMotorSpeed(leftDrive,50);
    setMotorSpeed(rightDrive,35);
  }
  else{
    setMotorSpeed(leftDrive,50);// go straight
    setMotorSpeed(rightDrive,50);
   }
  }
}
```

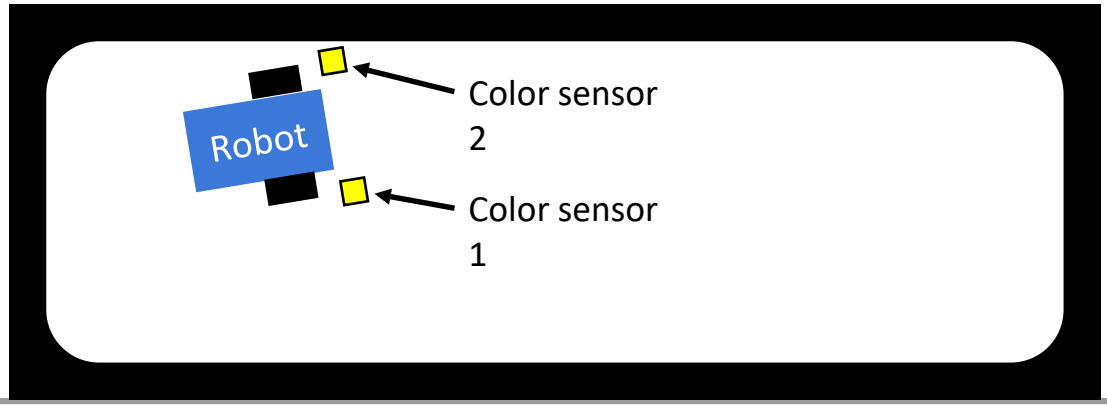
Program: ImprovedLineFollow.c

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

```
task main()
{
  while(true){
  if(getColorValue(leftEdge)<100){//if dark
    setMotorSpeed(leftDrive,50);
    setMotorSpeed(rightDrive,35);
  }
  else{
    setMotorSpeed(leftDrive,35);
    setMotorSpeed(rightDrive,50);
  }
  }
}
```

Line following to the corner

```
task main()
{
  wait1Msec(1000);//let sensor initialize
  while(getColorValue(rightEdge)>100){
  if(getColorValue(leftEdge)<100){//if dark
    setMotorSpeed(leftDrive,50);
    setMotorSpeed(rightDrive,35);
  }
  else{
    setMotorSpeed(leftDrive,35);
    setMotorSpeed(rightDrive,50);
  }
  }
  setMotorSpeed(leftDrive,0);
  setMotorSpeed(rightDrive,0);
}
```

Program: LineFollowStop.c

YouTube:
https://youtu.be/1z_GVJNjD94

Task 3

Line following a given distance

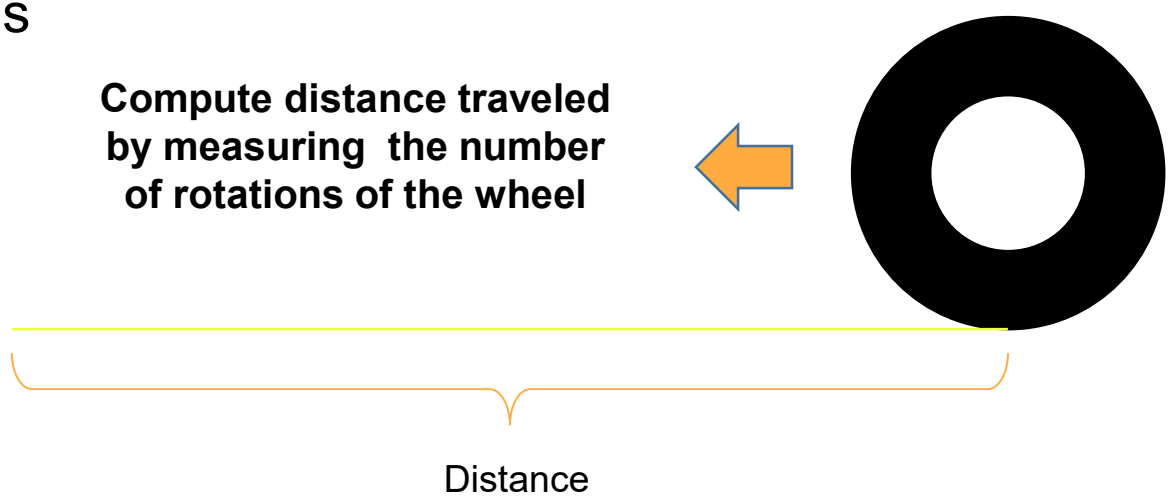
Line following a given distance

- Approach
  - Modify LineFollowZZStop.ev3 to stop when the robot travels a given distance
- Tools needed
  - Line following
  - Measure distance traveled

Measuring Distances

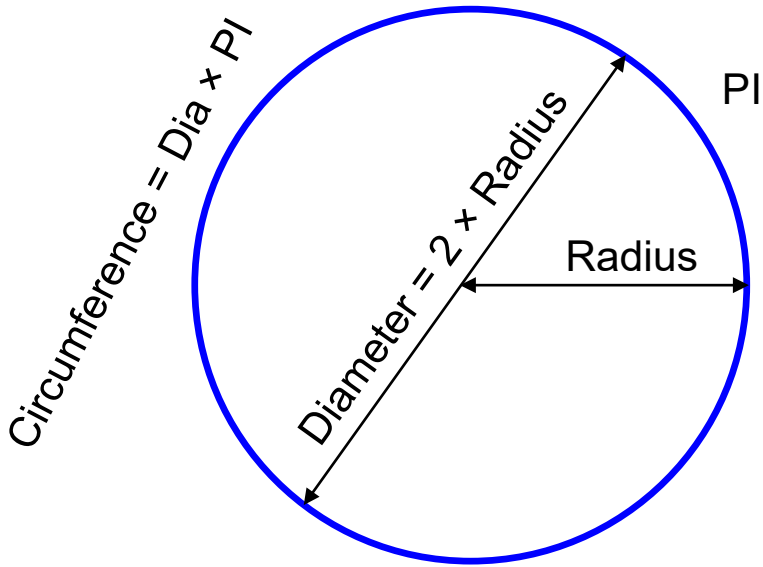
- How do we measure distance traveled?
- Let's determine how far the robot travels moving forward for 2 seconds

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



Measuring Distances

- Use the wheel geometry



How can we use this information?

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 (# Rotations)

Program: MeasureDistance.ev3

YouTube:
https://youtu.be/r0psO1aK0ME

```
task main()
{
  resetMotorEncoder(leftDrive);
  setMotorSpeed(leftDrive,50);
  setMotorSpeed(rightDrive,50);
  wait1Msec(2000);
  setMotorSpeed(leftDrive,0);
  setMotorSpeed(rightDrive,0);
  displayTextLine(1, "Distance CM: %d", getMotorEncoder(leftDrive)/360*20);
  waitUntil(getTouchLEDValue(touchLED)==1);
}
```

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)
    - Solve for (# Rotations)

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

$$(\#\ \text{Rotations}) = \frac{30\ \text{cm}}{(5.5\ \text{cm}) \times (\text{PI})} = 1.74\ \text{rotations}$$

Line following a given distance

```
task main()
{
  resetMotorEncoder(leftDrive);
  while(getMotorEncoder(leftDrive)<1000){
  if(getColorValue(leftEdge)<100){//if dark
    setMotorSpeed(leftDrive,50);
    setMotorSpeed(rightDrive,35);
  }
  else{ //not dark
    setMotorSpeed(leftDrive,35);
    setMotorSpeed(rightDrive,50);
  }
  }
  setMotorSpeed(leftDrive,0);
  setMotorSpeed(rightDrive,0);
}
```

Program: LineFollowDistance.c

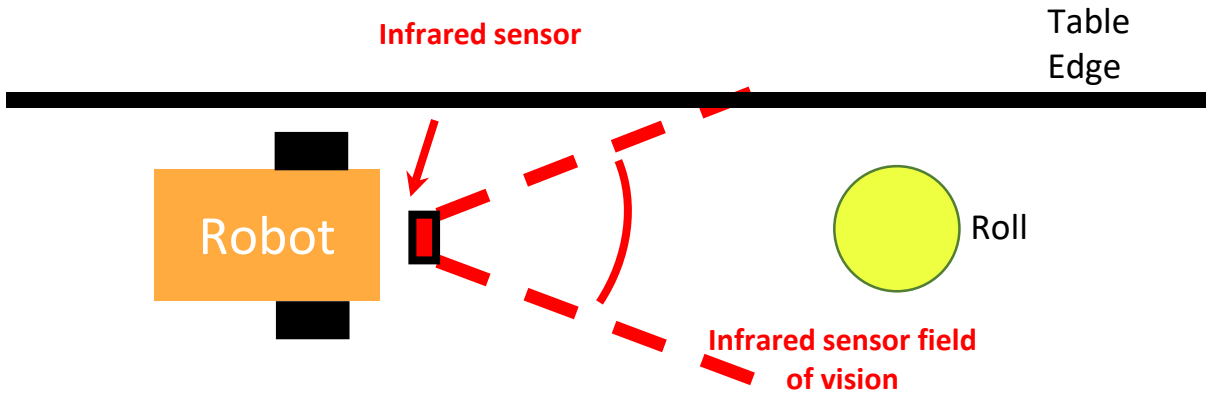
YouTube:
https://youtu.be/K1suXAzZuCA

Task 4

Finding a Roll

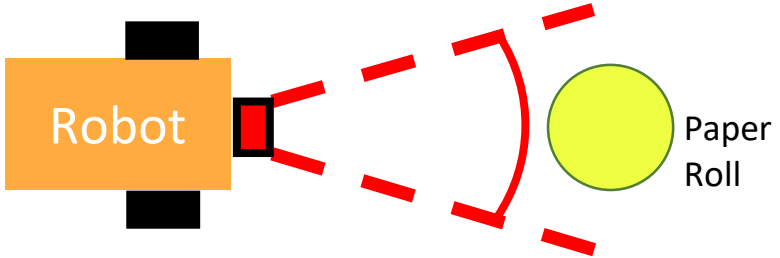
Finding a Rolls

- 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



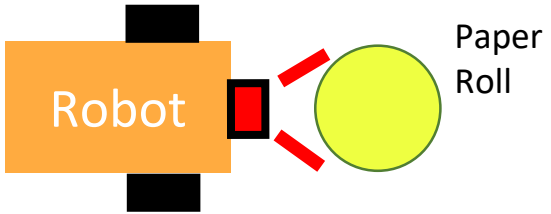
Finding a Paper Roll

- 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

```
task main()
{
  wait1Msec(1000);//let sensor initialize
  while(getColorValue(object)<200)
  if(getColorValue(leftEdge)<100){
    setMotorSpeed(leftDrive,50);
    setMotorSpeed(rightDrive,35);
  }
  else{
    setMotorSpeed(leftDrive,35);
    setMotorSpeed(rightDrive,50);
  }
  }
  setMotorSpeed(leftDrive,0);
  setMotorSpeed(rightDrive,0);
}
```

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.

Program: LineFollowObject.c

YouTube: https://youtu.be/-zrYAbNv5jI

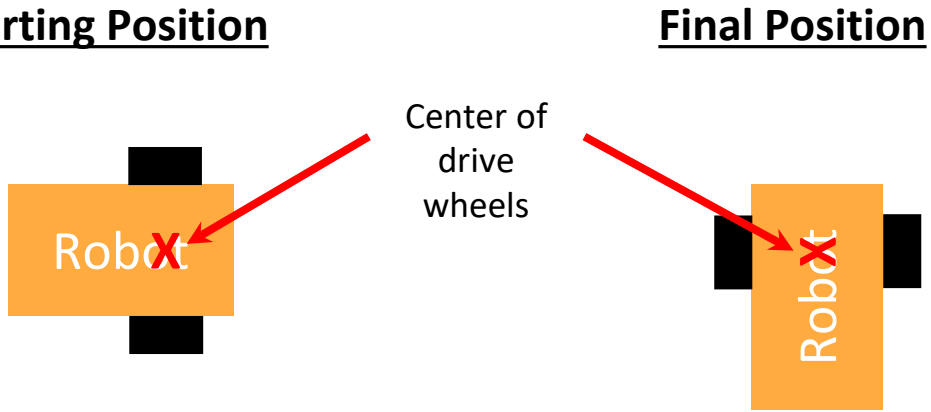
Task 5
Turning the robot

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



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

```
task main()
{

  // Initialize motor encoder
  resetMotorEncoder(rightMotor);

  // Start spinning left
  setMotorSpeed(leftMotor, -20);
  setMotorSpeed(rightMotor, 20);

  // Stop when robot spun 90 degrees
  while( (getMotorEncoder(rightMotor) / 360.0) < 0.65 )
  {
  }

  // Stop robot
  setMotorSpeed(leftMotor, 0);
  setMotorSpeed(rightMotor, 0);
}
```

Loop until the desired distance is traveled.

0.65 rotations was found using trail and error

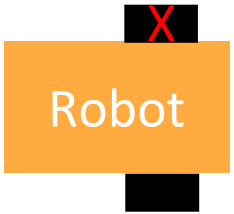
Program: Spin.c

YouTube: https://youtu.be/AUqXzsnf0NI

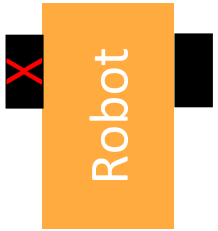
90 Degree Swing

- 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

```
task main()
{

  // Initialize motor encoder
  resetMotorEncoder(rightMotor);

  // Start swing motion to the left
  setMotorSpeed(leftMotor, 0);
  setMotorSpeed(rightMotor, 20);

  // Stop when robot spun 90 degrees
  while( (getMotorEncoder(rightMotor) / 360.0) < 1.35 )
  {
  }

  // Stop robot
  setMotorSpeed(leftMotor, 0);
  setMotorSpeed(rightMotor, 0);
}
```

Loop until the desired distance is traveled.

01.35 rotations was found using trail and error

Program: Swing.c

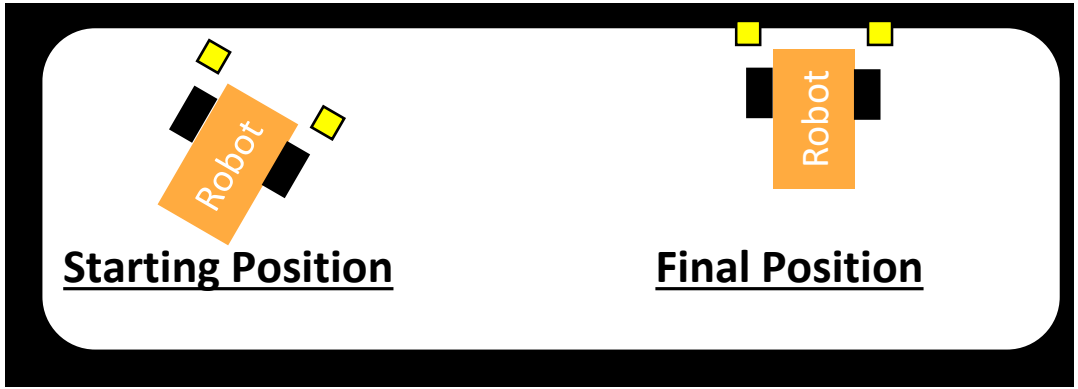
YouTube:
https://youtu.be/eQMffmd9zWM

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

```
task main()
{
//Turn on motors at 20%
 setMotorSpeed(leftMotor,20);
 setMotorSpeed(rightMotor,20);
 //Wait until left sensor reaches edge
 while(getColorReflected(leftColor)>20){
 }
 //Stop left side
 setMotorSpeed(leftMotor,0);
 //Wait unil right sensor reaches edge
 while(getColorReflected(rightColor)>20){
 }
 //Stop right side
 setMotorSpeed(rightMotor,0);
}
```

Go forward

Stop when left color sensor reaches edge

Power only the right side until the right color sensor reaches the edge

Aligning the robot to an edge- either sensor first

```
task main()
{
//Turn on motors at 20%
 setMotorSpeed(leftMotor,20);
 setMotorSpeed(rightMotor,20);
 //Wait until either sensor reaches edge
 while(getColorReflected(leftColor)>20&getColorReflected(rightColor)>20){
 }
 if(getColorReflected(leftColor)<20){
 //Stop left side
 setMotorSpeed(leftMotor,0);
 //Wait until right sensor reaches edge
 while(getColorReflected(rightColor)>20){
 }
 //Stop right side
 setMotorSpeed(rightMotor,0);
}
else{
  //stop right side
  setMotorSpeed(rightMotor,0);
 //Wait unil right sensor reaches edge
 while(getColorReflected(leftColor)>20){
 }
 //Stop right side
 setMotorSpeed(leftMotor,0);
 }
}
```

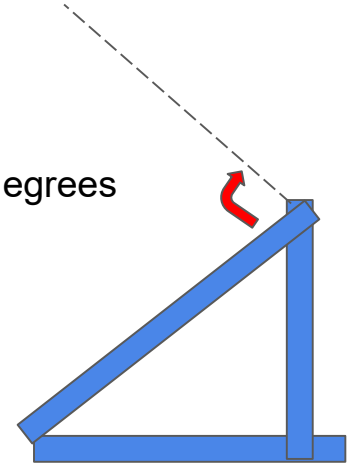
Program: Align2.c **YouTube**: https://youtu.be/Jq0L2ekQARk

Task 7

Manipulating Paper Rolls
Moving the Arm and Claw

How to Control the Arm and Claw

- Time
- Rotation(encoder degrees)



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

```
task main()
{
 setMotorSpeed(arm,100);
 sleep(2000);
 setMotorSpeed(arm,0);
}
```

Raise Arm

```
task main()
{
 setMotorSpeed(arm,-100);
 sleep(2000);
 setMotorSpeed(arm,0);
}
```

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

```
task main()
{
  //set zero
  resetMotorEncoder(arm);
  //raise arm
  setMotorSpeed(arm,100);
  while(getMotorEncoder(arm)<1000){
  }
  setMotorSpeed(arm,0);
  //pause
   sleep(1000);
   // lower arm
   setMotorSpeed(arm,-50);
 while(getMotorEncoder(arm)>0){
  }
  setMotorSpeed(arm,0);
}
```

Raise Arm

Lower Arm

Moving the Claw

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

```
task main()
{
  //set zero
  resetMotorEncoder(claw);
  //close claw
  setMotorSpeed(claw,100);
  sleep(1000);
  setMotorSpeed(claw,0);
  //pause
   sleep(1000);
   // open claw
   setMotorSpeed(claw,-50);
  while(getMotorEncoder(claw)>0){
  }
  setMotorSpeed(claw,0);
}
```

Use time to close

Use rotation to open

Combine Arm and Claw Movement

```
task main()
{
  //set zero
  resetMotorEncoder(arm);
  resetMotorEncoder(claw);
  //close claw
  setMotorSpeed(claw,100);
  sleep(1000);
  setMotorSpeed(claw,0);
  //raise arm
  setMotorSpeed(arm,100);
  while(getMotorEncoder(arm)<1000){
  }
  setMotorSpeed(arm,0);
  //pause
   sleep(1000);
   // lower arm
   setMotorSpeed(arm,-50);
 while(getMotorEncoder(arm)>0){
  }
  setMotorSpeed(arm,0);
  // open claw
   setMotorSpeed(claw,-50);
  while(getMotorEncoder(claw)>0){
  }
  setMotorSpeed(claw,0);
}
```

Program: ArmAndClaw.c

YouTube: https://youtu.be/bgfusgzjxiM

Task 8

Building Functions

Functions

- Solving the Robofest Game challenge will typically require a fairly large program (over 100 lines is not unreasonable)
- Very large programs can be difficult to understand, navigate and use
- To alleviate this issue, the RobotC software can be used tocreate custom functions that can replace sections of your program

Functions

- 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, then stop
- The code may look like this

```
setMotorSpeed(leftDrive,50);//start robot
setMotorSpeed(rightDrive,50);
while(getColorValue(leftEdge)>100){//wait for dark
}
setMotorSpeed(leftDrive,0);// stop robot
setMotorSpeed(rightDrive,0);
```

- Functions will allow us to convert this to a single line

```
toEdge();
```

Function Set up

```
void toEdge(){

  setMotorSpeed(leftDrive,50);//start robot
  setMotorSpeed(rightDrive,50);
  while(getColorValue(leftEdge)>100){//wait for dark
  }
  setMotorSpeed(leftDrive,0);// stop robot
  setMotorSpeed(rightDrive,0);
}


task main()
{
  wait1Msec(1000);
  toEdge();


}
```

Define function above the task main

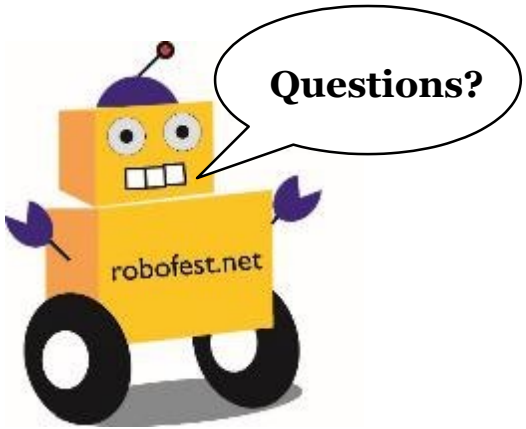
Use task in the main program

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 Functions

LAWRENCE TECHNOLOGICAL UNIVERSITY ROBOFEST

Little Robots, Big Missions



robofest@LTU.edu
LTU Computer Science

LAWRENCE TECHNOLOGICAL UNIVERSITY ROBOFEST

2021 Workshops

Sponsored by

Lawrence Technological University

Computer Science