PRIME LESSONS

By the Makers of EV3Lessons

LINE FOLLOWER

BY SANJAY AND ARVIND SESHAN

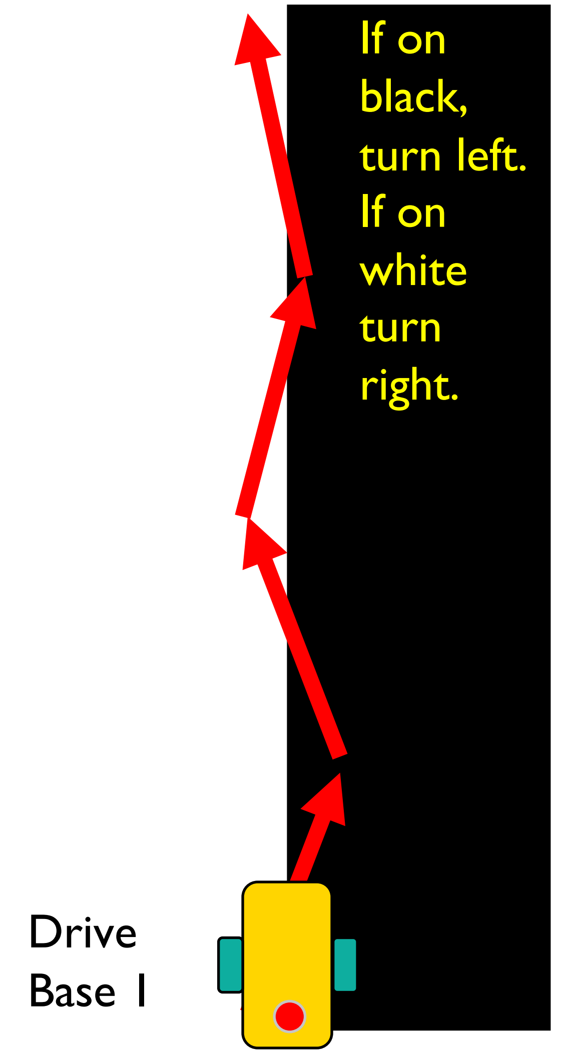
This lesson uses SPIKE 3 software

LESSON OBJECTIVES

Learn how to get a robot to follow a line using Color Mode or Reflected Light Mode on the SPIKE Prime Color Sensor

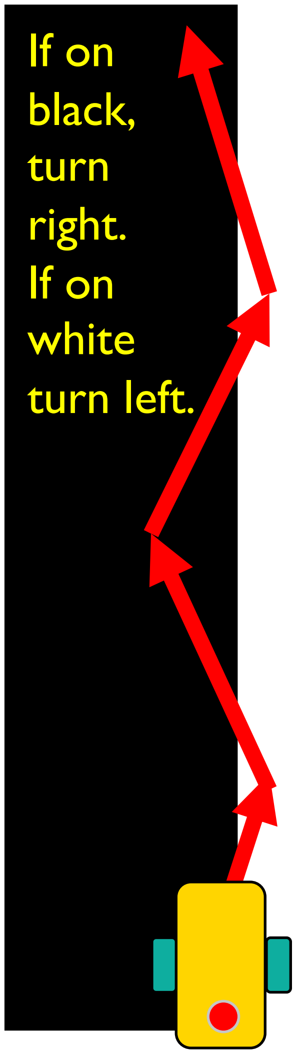
Learn how to combine sensors, loop, and conditionals

ROBOTS FOLLOW THE EDGE OF THE LINE

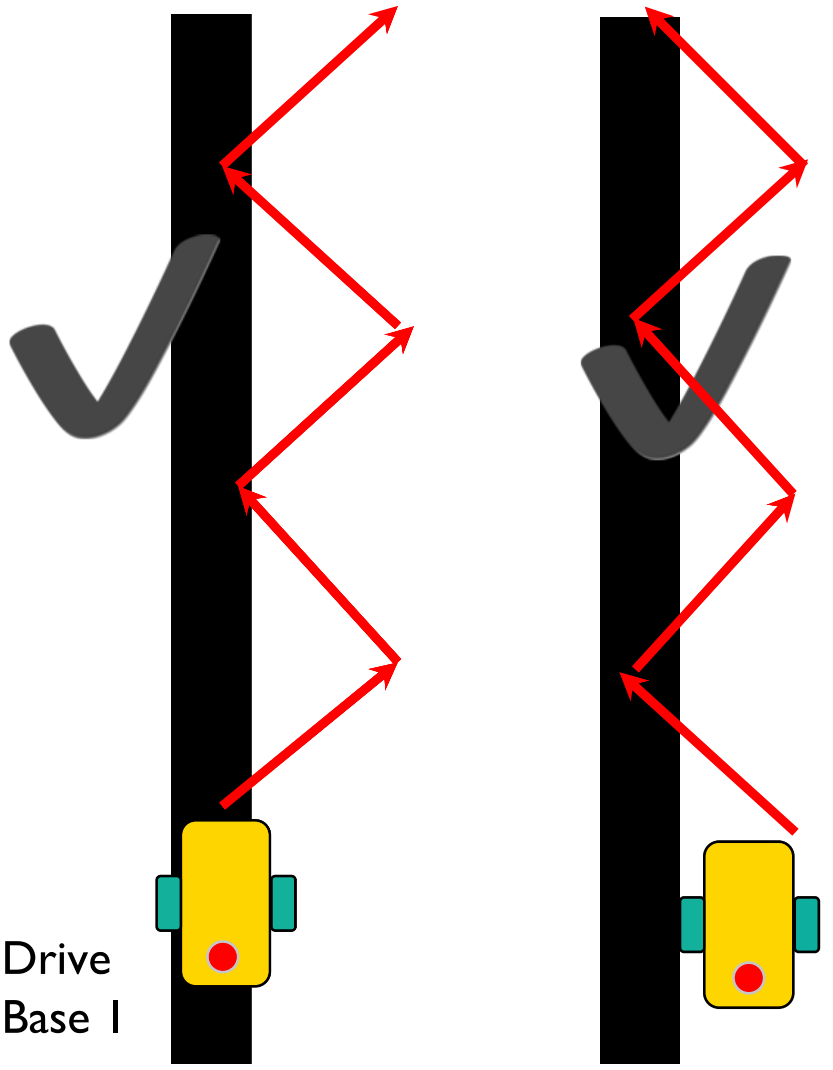


The robot has to choose which way to turn when the color sensor sees a different color.

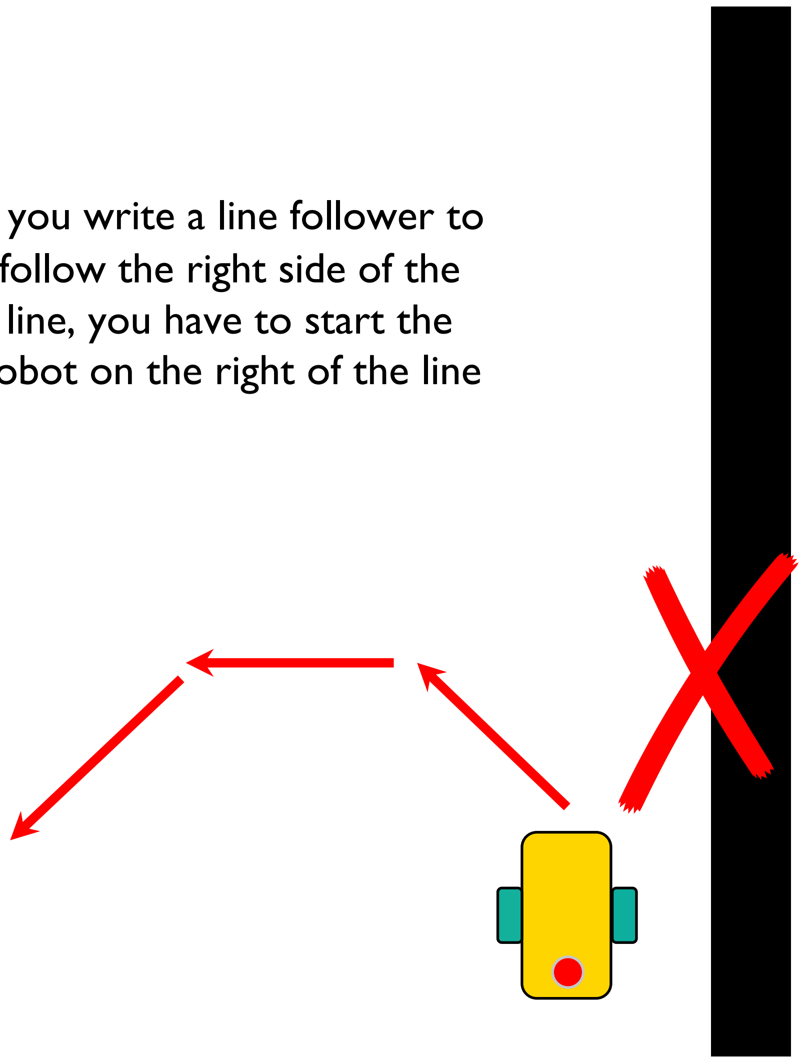
The answer depends on what side of the line you are following!



WHICH SIDE OF THE LINE SHOULD YOU START ON



If you write a line follower to follow the right side of the line, you have to start the robot on the right of the line



CHALLENGE: FOLLOW A LINE

Write a program that follows the right edge of the line

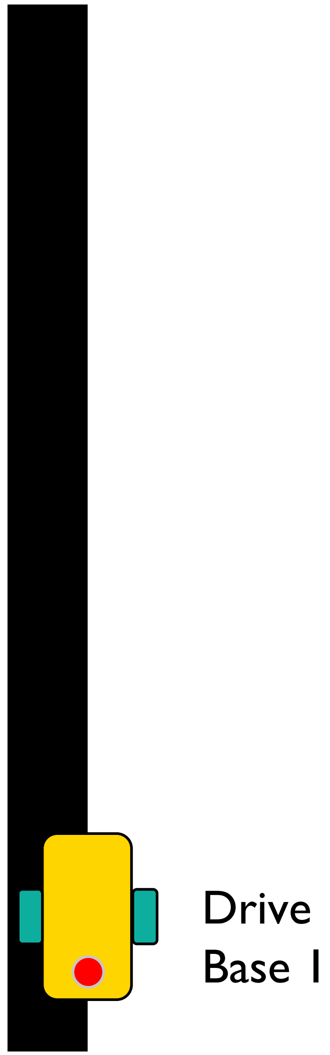
If your sensor sees black, turn right

If your sensor sees white, turn left

Use a conditional to make that decision

Repeat the line follower forever

Use Color Mode or Reflected Light Mode



Note: To line follow with the Advanced Driving Base (ADB) in Color Mode you will have to make a modification to the design because the color sensor does not recognize black at the height in the original build instructions. See our Color Sensor lesson.

TWO WAYS TO TURN

A previous lesson, "Turning with the Gyro" explained two motor pair functions to make the robot turn. Please refer to that lesson for details.

1. You can use motor_pair.move and adjust the steering value. This lesson will use steering.

Change steering value here. A value of 0 moves straight

```
motor_pair.move(pair, steering)
```

1. You can use motor_pair.move_tank and input different velocity values for the left and right motors. You can try this out on your own.

Change velocity values here. Same velocity values moves straight

```
motor_pair.move_tank(pair, left_velocity, right_velocity)
```

LINE FOLLOWER – COLOR & REFLECTED MODE

```python
from hub import port
import motor_pair, color_sensor, runloop

# Constants for Drive Base 1
motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)

# Follow the right side of black line (Black-White edge). NOTE: Our test was run on a black-white mat.
# If your mat has many colors, you will have to lower the threshold to avoid other colors.
# To follow a White-Black edge, change the IF condition from < 50 to > 50
# To use color mode, import color, and use condition:
# if (color_sensor.color(port.A) == color.BLACK)
async def line_follow_forever():
    while (True):
        if (color_sensor.reflection(port.A) < 50): # sensor is on Black. Lower threshold as needed for your case.
            # Turn right, i.e. away from Black
            motor_pair.move(motor_pair.PAIR_1, 30, velocity = 300)
        else: # sensor is on white
            # Turn left, i.e. towards Black
            motor_pair.move(motor_pair.PAIR_1, -30, velocity = 300)

async def main():
    await line_follow_forever()

runloop.run(main())
```

EXTENSION - CHANGING EXIT CONDITIONS

In FLL, you typically do not want to line follow forever. You may want to stop under some conditions, some of which can be:

1. Your ultrasonic sensor detected something
2. Your force sensor was pressed
3. You have a second color sensor on your robot that sensed a marker on the mat. This is extremely useful in FLL.
4. You want to line follow for an approximate distance.

    Hint: you can reset an individual motor's relative position and then stop when it crosses a value that maps to the distance you want to follow. Be mindful of clockwise/counterclockwise motor rotation

Combine this lesson with the Loops lesson to solve this problem.

LINE FOLLOW UNTIL SECOND SENSOR SEES BLACK

```
from hub import port
import motor_pair, color_sensor, runloop, sys

motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)

# follow right side of black line (Black-White edge) until second sensor sees Black
# Test mat has only Black and White colors. Adjust threshold of 50 to a lower value as you need.
async def line_follow_until_line():
    # Drive Base 1 is modified to have a second color sensor at port B.
    # Follow line until sensor B sees black
    while (color_sensor.reflection(port.B) > 50): # Adjust threshold as needed.
        if (color_sensor.reflection(port.A) < 50): # sensor is on Black. Adjust threshold as needed.
            # Turn right, i.e. away from Black
            motor_pair.move(motor_pair.PAIR_1, 30, velocity = 300)
        else: # sensor is on white
            # Turn left, i.e. towards Black
            motor_pair.move(motor_pair.PAIR_1, -30, velocity = 300)

async def main():
    await line_follow_until_line()
    sys.exit("Stopping")

runloop.run(main())
```

LINE FOLLOW FOR APPROXIMATE DISTANCE

```
from hub import port
import motor, motor_pair, color_sensor, runloop, sys

motor_pair.pair(motor_pair.PAIR_1, port.C, port.D) # Drive base 1 (DB1)
WHEEL_CIRCUMFERENCE = 17.5 # cm – wheel size for DB1

# follow right side of black line (Black-White edge) until distance is covered.
async def line_follow_for_distance_cm(distance_cm):
    # Calculate the number of degrees to turn to cover the desired distance.
    # See lesson on More Accurate Turns for explanation.
    motor_degrees = int((distance_cm/WHEEL_CIRCUMFERENCE) * 360)
    # Use motor D for DB1 because it moves clockwise and the degrees count up.
    motor.reset_relative_position(port.D, 0)
    while (motor.relative_position(port.D) < motor_degrees):
        if (color_sensor.reflection(port.A) < 50): # sensor is on Black. Adjust threshold as needed if this is too high
            motor_pair.move(motor_pair.PAIR_1, 30, velocity = 300) # Turn right
        else: # sensor is on white
            motor_pair.move(motor_pair.PAIR_1, -30, velocity = 300) # Turn left

async def main():
    await line_follow_for_distance_cm(70)
    sys.exit("Stopping")

runloop.run(main())
```

CREDITS

This lesson was created by Sanjay and Arvind Seshan for Prime Lessons

Additional contributions by FLL Share & Learn community members.

More lessons are available at www.primelessons.org