Lego Robotics Camp

Day 3: Using Our Senses

Review

- Yesterday we:
  - Learned about **conditionals** and **loops**
  - Design simple algorithms for parking our robots

```
if condition1 is true:
     do action 1
elif condition2 is true:
     do action 2
else:
     do action 3
```

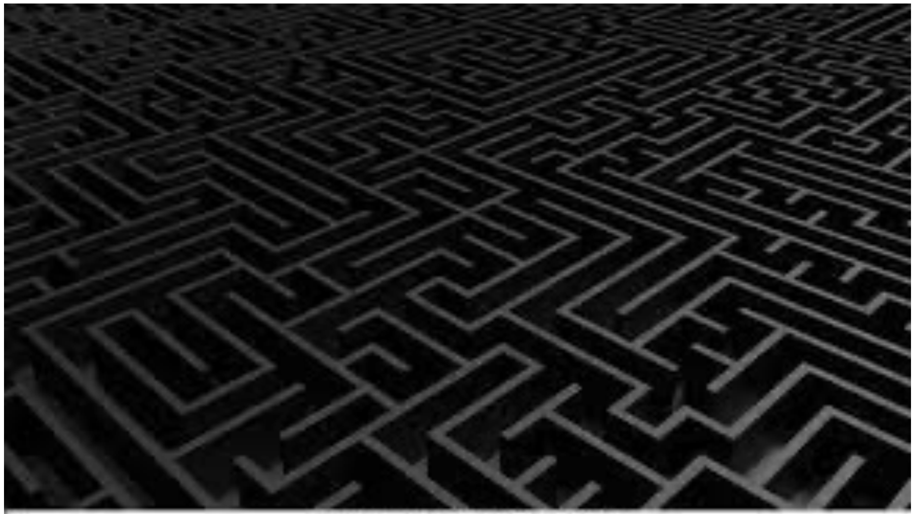
```
while condition1 is true:

     do action1
```

Today's Plan

- Today we will:
  - Learn about using sensors and handling simple events
  - Practice using conditionals and loops
  - Design simple algorithms for solving mazes

Solving Mazes

- We are going to go outside and practice solving mazes
- …without using our senses!
- Like solving a maze in the dark!
- Two of you will be blindfolded and will not be able to see the maze at all
- The other two have 5-10 minutes to determine the instructions that your teammates will use to solve the maze



shutterstock.com · 1791169847

Solving Mazes

- Rules:
  - You cannot speak to the maze runners after they begin “executing” their instructions
  - Can only use basic instructions (for now): go straight for 5 steps, turn right, etc
  - The maze runners will execute one instruction at a time
  - Maze runner will start at the START line. The goal is to get as close to END as possible.
  - Maze runners cannot remove blindfold until instructed to do so ☺

Solving Mazes Lessons Learned

- Which was easier: solving mazes with or without loops?
- Which was easier: solving mazes with or without help from your senses?

- Today we are going to learn how to give our robots the capability to sense various aspects of their environment using **sensors**

BREAK

Using Sensors

Infrared

Ultrasonic

Touch

Gyro

Color

Infrared Sensor

- Mostly used for creating remotely-controlled robots
- Listens for infrared beacon from remote control

Ultrasonic Sensor

- Generates sound waves and reads their echoes to detect and **measure distance** from objects
- Very good accuracy

Touch Sensor

- Detect when the sensor's red button has been pressed or released
- Measures contact or “touch” with surfaces or objects

Color Sensor

- Measures color and darkness
- Measure intensity of reflected light
- Measure intensity of ambient light
- Detect any one of seven colors (black, blue, green, yellow, red, white, brown) or no color

Gyro Sensor

- Measures the robot's rotation and changes in its orientation

Our Robots

- Our goal is to make our robots solve (arbitrary) mazes of blocks
- Our robots will use touch, ultrasonic, and color (tomorrow) sensors

- Let’s start with the touch sensor

Touch Sensor

- What might be use this sensor for?
- How can we use it to help our robot solve a maze?

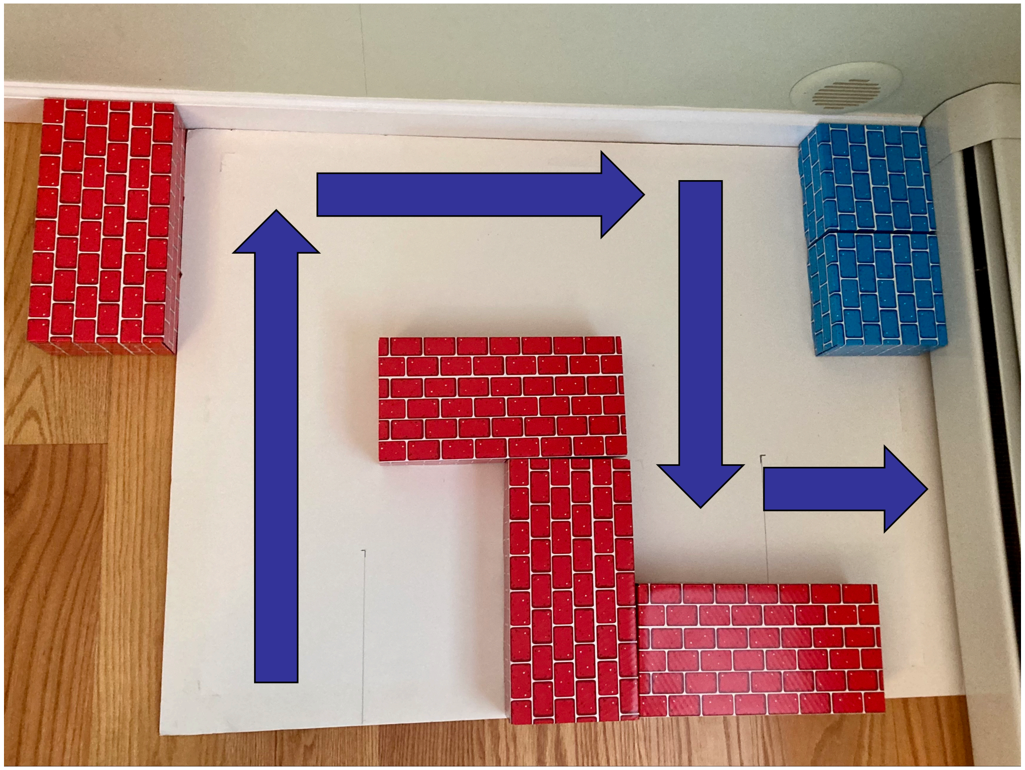
- Think about how you can find your way out of this:

Touch Sensor

Maze algorithm

- Walk until you hit a wall!
- Then what?
- Turn in one direction
- Repeat!
- Eventually (maybe?) you'll find your way out

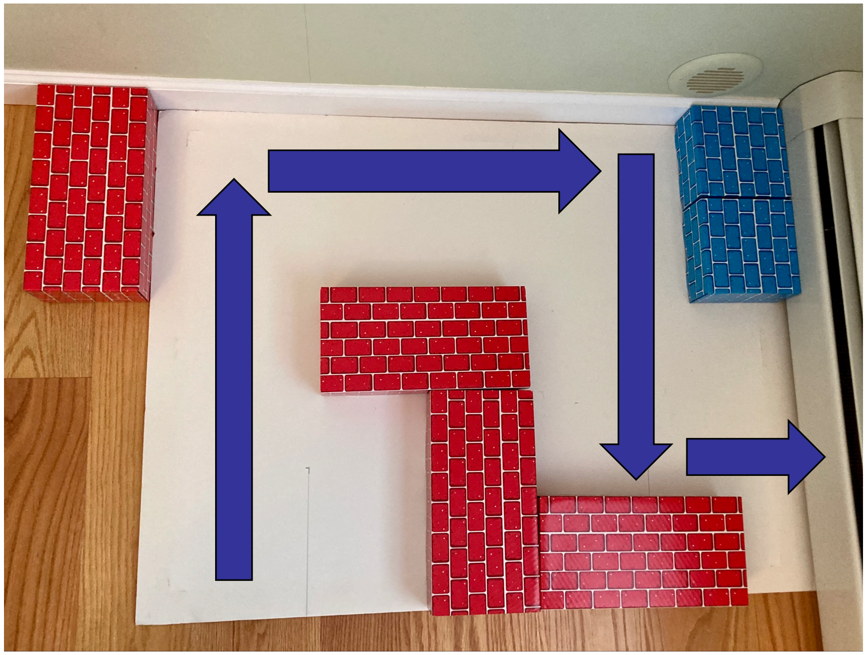
Touch Sensor



- Let’s try this with my robot
- What is the general algorithm for solving a maze with our robots?
- (Remember our CT pillars)
- Our robots don’t have eyes!
- Imagine you are in a dark room trying to find your way out!

bump_maze

1. Go forward until you bump a wall
2. Back up a little bit
3. Turn right
4. Go forward until you bump a wall
5. Back up a little
6. Turn right
7. Go forward until you bump a wall
8. Back up a little
9. Turn left
10. Go forward



bump_maze

1. Go forward until you bump a wall
2. Back up a little bit
3. Turn right
4. Go forward until you bump a wa
5. Back up a little
6. Turn right
7. Go forward until you bump a wall
8. Back up a little
9. Turn left
10. Go forward

We know how to do this:

```
robot.straight(200)  or
robot.drive(100,0)

robot.turn(90)
```

bump_maze

1. Go forward until you bump a wall
2. Back up a little bit
3. Turn right
4. Go forward until you bump a wa
5. Back up a little
6. Turn right
7. Go forward until you bump a wall
8. Back up a little
9. Turn left
10. Go forward

How do we go forward until we bump a wall?

We want to go forward until the touch sensor is pressed

We want to go forward **while** touch sensor is not pressed.

bump_maze

1. Go forward until you bump a wall
2. Back up a little bit
3. Turn right
4. Go forward until you bu
5. Back up a little
6. Turn right
7. Go forward until you bump a wall
8. Back up a little
9. Turn left
10. Go forward

```
touch_sensor = TouchSensor(Port.S1)
while not touch_sensor.pressed():
    robot.drive(200,0)
    wait(10) #wait 10ms, then repeat
```

bump_maze

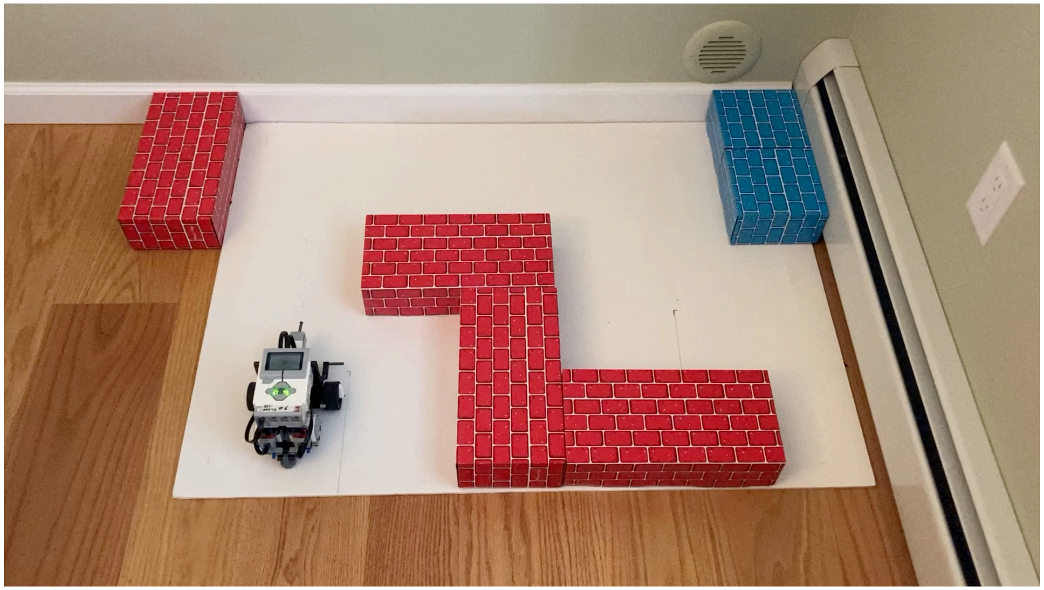
```
#Initialize the touch sensor
touch_sensor = TouchSensor(Port.S1)
while not touch_sensor.pressed():
        robot.drive(200,0)
        wait(10)
robot.straight(-20)
robot.turn(90)
while not touch_sensor.presse
        robot.drive(200,0)
        wait(10)
robot.straight(-20)
robot.turn(90)
while not touch_sensor.pressed():
        robot.drive(200,0)
        wait(10)
robot.straight(-20)
robot.turn(-90)
robot.straight(300)
```

Put it all together:

```
while not touch_sensor.pressed():
        robot.drive(200,0)
        wait(10)
robot.straight(-20)  #back up
robot.turn(90)       #turn right
```

bump_maze

```
#Initialize the touch sensor
touch_sensor = TouchSensor(Port.S1)
while not touch_sensor.pressed():
       robot.drive(200,0)
       wait(10)
robot.straight(-20)
robot.turn(90)
while not touch_sensor.pressed():
       robot.drive(200,0)
       wait(10)
robot.straight(-20)
robot.turn(90)
while not touch_sensor.pressed():
       robot.drive(200,0)
       wait(10)
robot.straight(-20)
robot.turn(-90)
robot.straight(300)
```



Touch Sensor

- The touch sensor allowed us to detect walls by bumping or touching
- Bumping/touch triggers an **event** in our program which we can react to
- Works, but isn't perfect
- What other sensors might be even better?

Ultrasonic Sensor

- Detect when we are close to a wall WITHOUT running into it
- Follow similar algorithm, but measure distance to wall instead of waiting for sensor to be pressed
- What are we sensing?
- What is the "event?"

ultrasonic_maze

1. Go forward until you get close to a wall
2. Turn right
3. Go forward until you get close to a wall
4. Turn right
5. Go forward until you get close to a wall
6. Turn left
7. Go forward

ultrasonic_maze

1. Go forward until you get close to a wall
2. Turn right
3. Go forward until you get clo a
4. Turn right
5. Go forward until you get close to a
6. Turn left
7. Go forward

How do we do this using the ultrasonic sensor?

Go forward until the distance to the wall is sufficiently small

Go forward **while** the distance to the wall is small

ultrasonic_maze

1. Go forward until you get close to a wall
2. Turn right
3. Go forward until you get
4. Turn right
5. Go forward until you get close
6. Turn left
7. Go forward

```
us_sensor = UltrasonicSensor(Port.S1)
```

To just measure the distance:

```
dist = us_sensor.distance()
```

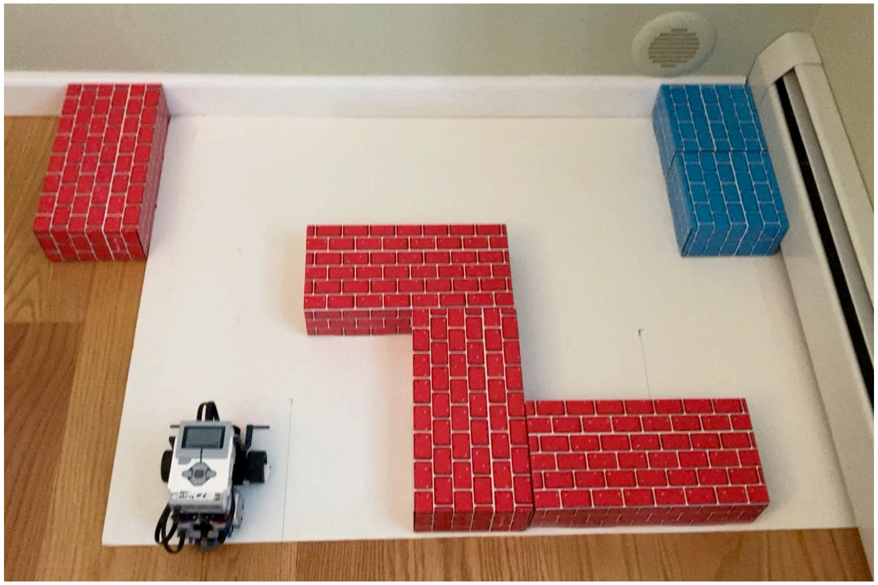
ultrasonic_maze

1. Go forward until you get close to a wall
2. Turn right
3. Go forward until you get
4. Turn right
5. Go forward until you get close
6. Turn left
7. Go forward

```
us_sensor = UltrasonicSensor(Port.S1)

while us_sensor.distance() > 50:
    robot.drive(100,0)
    wait(10)
```

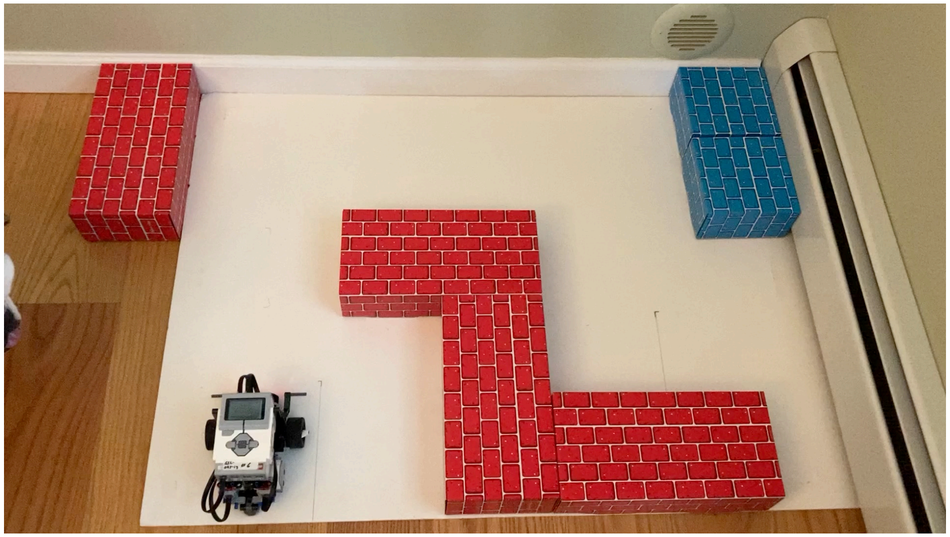
ultrasonic_maze

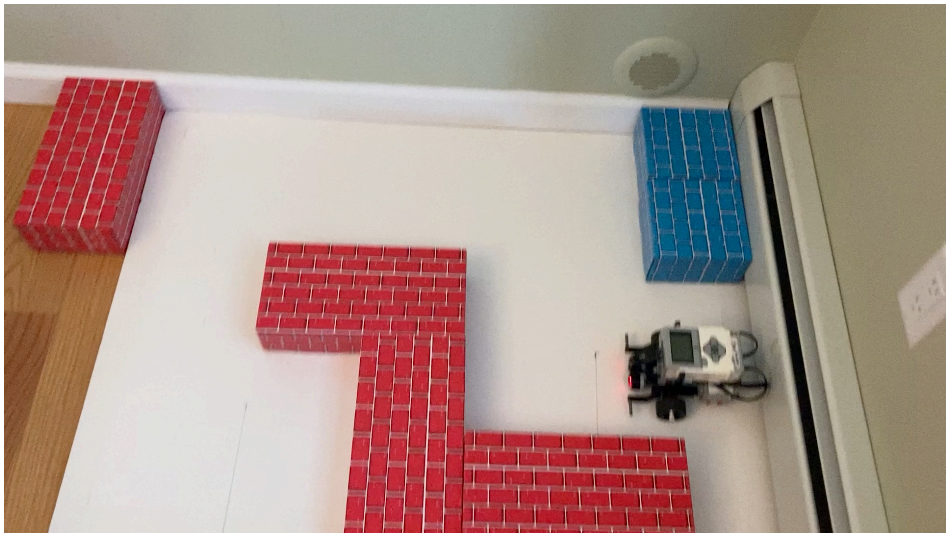
1. Go forward until you get close to a wall
2. Turn right
3. Go forward until you get close to a wall
4. Turn right
5. Go forward until you get close to a wall
6. Turn left
7. Go forward



Lab today

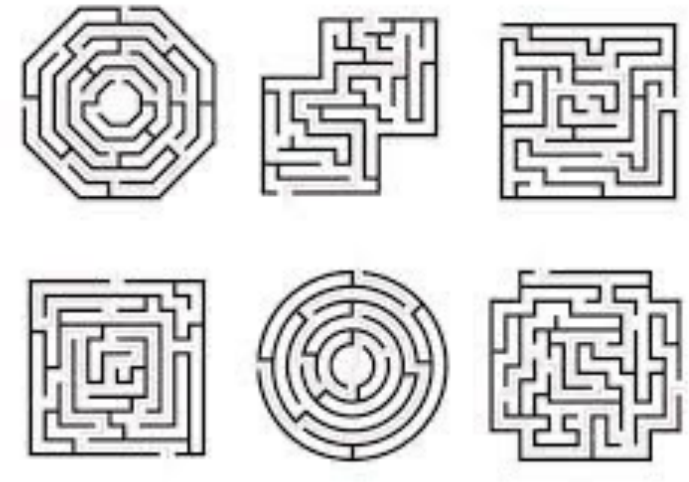
- What if we wanted to solve any 3-turn maze (not just the one we've been testing today)?
- What algorithm could we use?
- (Ignore the godzilla dog)





Lab today

- What if we wanted to solve any 3-turn maze?
- What algorithm could we use?
    - Go forward until you get close to a wall
    - Turn 90 degrees
    - Measure distance
    - Turn 180 degrees
    - Measure distance
    - Go forward in direction of greatest distance
    - Repeat for each turn
    - You will need to use sensors, while loops, and if statements
    - You can work with a partner!
    - Start by adding the ultrasonic sensor to your robot



LUNCH BREAK