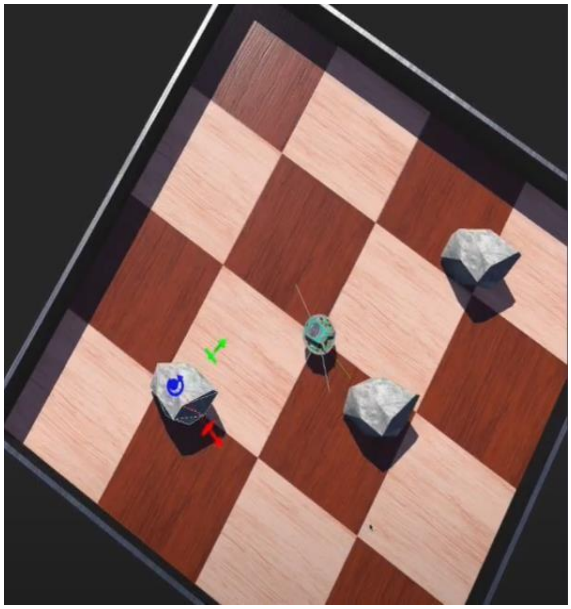
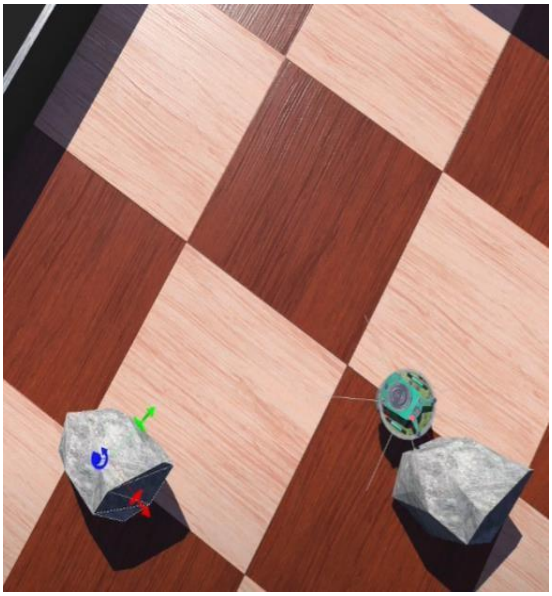
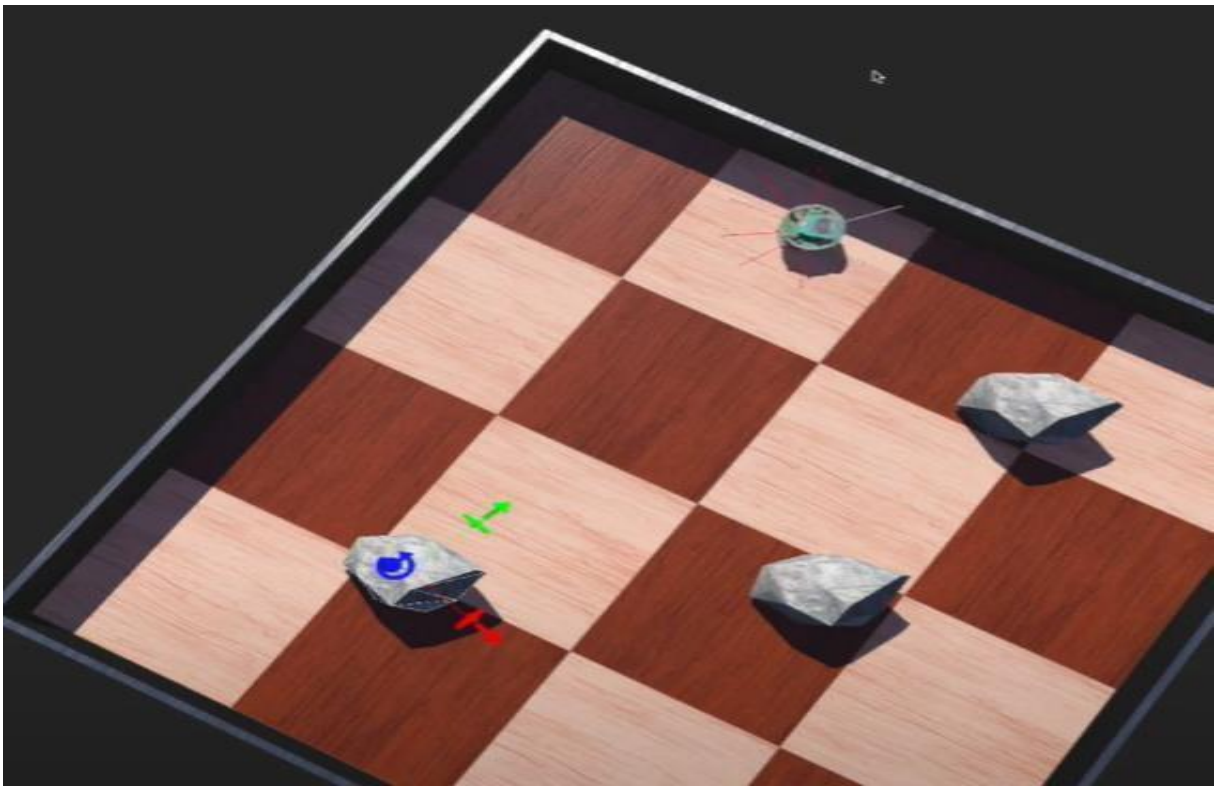
Obstacle Avoidance

```
from controller import Robot, Motor, DistanceSensor
TIME_STEP = 64 robot
= Robot()
left_motor = robot.getDevice("left wheel motor")
right_motor = robot.getDevice("right wheel motor")
left_motor.setPosition(float('inf'))
right_motor.setPosition(float('inf'))
left_motor.setVelocity(0) right_motor.setVelocity(0)
sensor_names = ["ps0", "ps1", "ps2", "ps3", "ps4", "ps5", "ps6", "ps7"]
ir_sensors = [robot.getDevice(name) for name in sensor_names] for
sensor in ir_sensors:     sensor.enable(TIME_STEP) def
adjust_motor_speed(sensor_values):
   left_speed = 3.0  # base speed
right_speed = 3.0  # base speed
   if sensor_values[0] > 100 or sensor_values[1] > 100:  # Obstacle to the left
left_speed -= 2.0  # turn right         right_speed += 2.0
   elif sensor_values[6] > 100 or sensor_values[7] > 100:  # Obstacle to the right
left_speed += 2.0  # turn left         right_speed -= 2.0     return left_speed,
right_speed while robot.step(TIME_STEP) != -1:
   sensor_values = [sensor.getValue() for sensor in ir_sensors]
left_speed, right_speed = adjust_motor_speed(sensor_values)
left_motor.setVelocity(left_speed)
right_motor.setVelocity(right_speed)
```

Output:







Line Follower

```
from controller import Robot, Motor, DistanceSensor
TIME_STEP = 32 robot = Robot()
left_motor = robot.getDevice("left wheel motor")
right_motor = robot.getDevice("right wheel motor")
left_motor.setPosition(float('inf'))
right_motor.setPosition(float('inf'))
left_motor.setVelocity(0) right_motor.setVelocity(0)
sensors = ["ir0", "ir1", "ir2", "ir3", "ir4", "ir5", "ir6", "ir7"]
ir_sensors = [robot.getDevice(sensor_name) for sensor_name in sensors] for
sensor in ir_sensors:     sensor.enable(TIME_STEP) def read_sensors():
   return [sensor.getValue() for sensor in ir_sensors] def
compute_control(sensor_values):
   set_point = 700     left_sensor =
sensor_values[2]     right_sensor =
sensor_values[5]     error =
left_sensor - right_sensor
   Kp = 0.005
Ki = 0.0
   Kd = 0.001
   P = error
   I = 0
   D = error - compute_control.prev_error if hasattr(compute_control, 'prev_error') else 0
pid_output = Kp * P + Ki * I + Kd * D     compute_control.prev_error = error     base_speed =
2.0
   left_speed = base_speed - pid_output
right_speed = base_speed + pid_output     return
left_speed, right_speed
while robot.step(TIME_STEP) != -1:     sensor_values
= read_sensors()
```

```
    left_speed, right_speed = compute_control(sensor_values)
left_motor.setVelocity(left_speed)
right_motor.setVelocity(right_speed)
```

Output: