Finite State Machine and Maze Programming

有限状态机和解迷宫编程

References
http://en.wikipedia.org/wiki/Finite-state_machine

Lecture 08

Yinong Chen

Roadmap

1. Maze Simulation 迷宫仿真
2. Maze Navigation Algorithms 解迷宫的算法
3. Communication between VIPLE and Different Robots
   VIPLE与不同机器人的通讯

The Project...

A person or an **autonomous robot** must navigate through a maze.

一个人或一个自主机器人必须在迷宫中航行。

The navigation problem is: solving a maze from the inside.

导航问题是：从内部解决迷宫

The algorithm receives its inputs gradually rather than all at once.

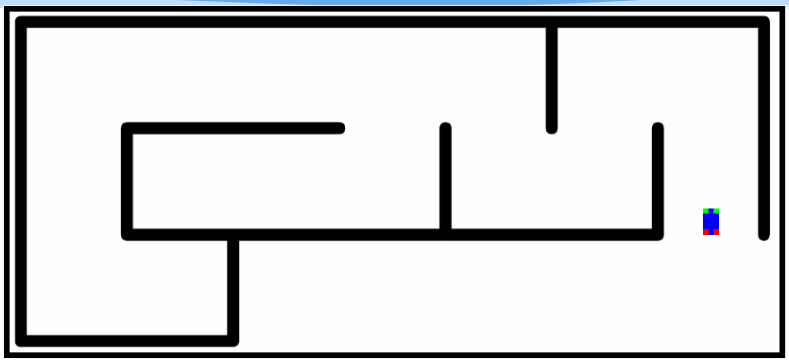
导航算法逐渐接收输入，而不是一次性接收。

It must make decisions based on this partial inputs.

它必须根据这部分输入做出决定。

ASU Web 2D Simulator: Right Wall Following
ASU Web二维模拟器：绕右墙

http://neptune.fulton.ad.asu.edu/VIPLE/Web2DSimulator/



Add a New Line | Remove a Line

Default: Forward

1. If Right sensor > 100 pixels

Then: Delayed Turn Right by 90 pixels

2. Else if Forward sensor <= 50 pixels

Then: Turn Left

Run

ASU Web 2D Simulator: Two-Distance Local Best

ASU Web二维模拟器：两距离局部最优

http://neptune.fulton.ad.asu.edu/VIPLE/Web2DSimulator/

Add a New Line | Remove a Line

Default: Forward

1. If Forward sensor > 100 pixels

Then: Move Forward

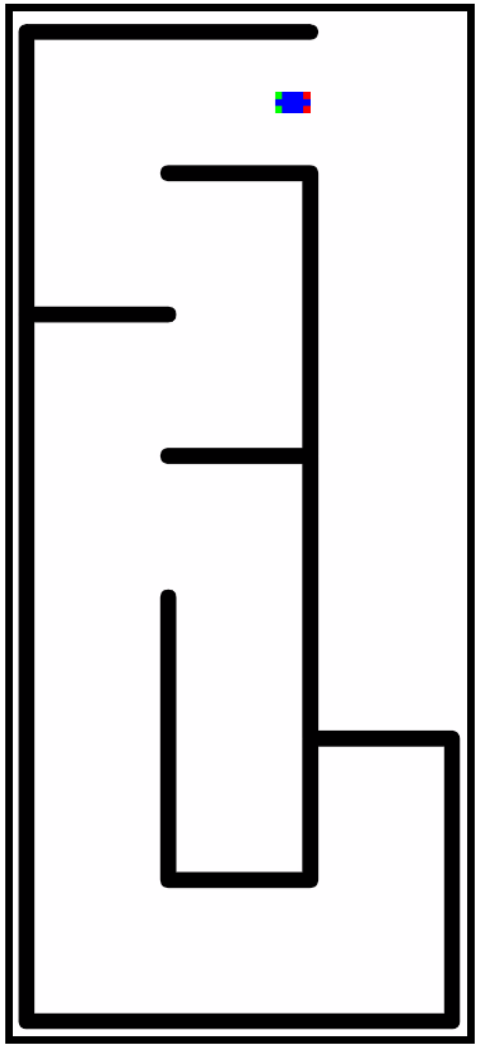
2. Else if Left - Right sensor > 0 pixels

Then: Turn Left

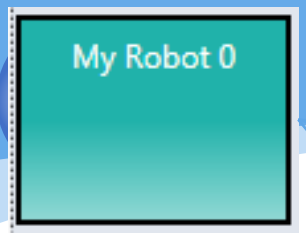
3. Else if Left - Right sensor <= 0 pixels

Then: Turn Right

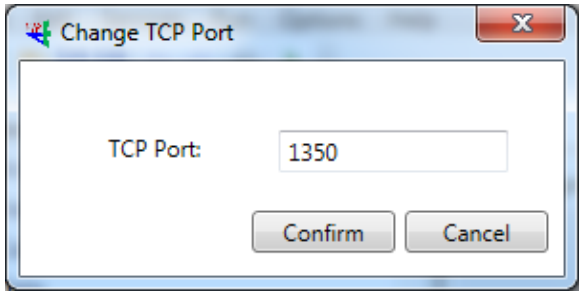
Run

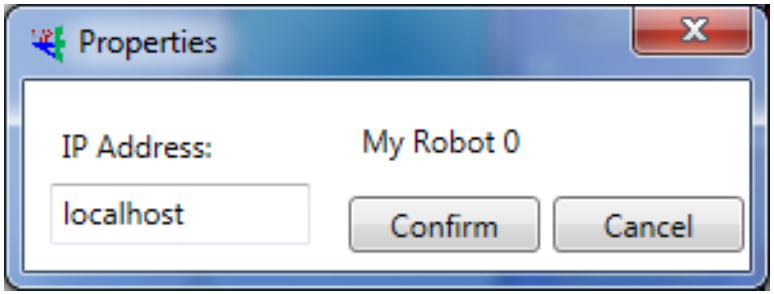


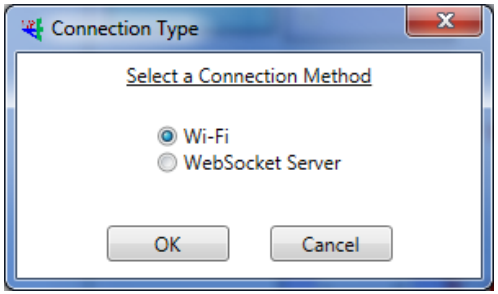
Unity Simulator Configuration
Unity 仿真器配置

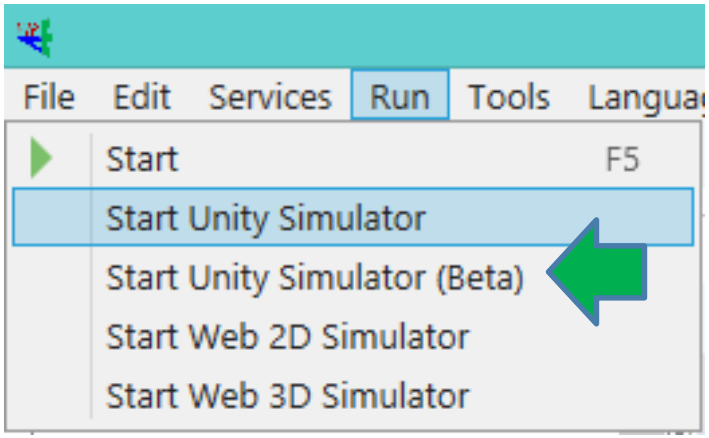


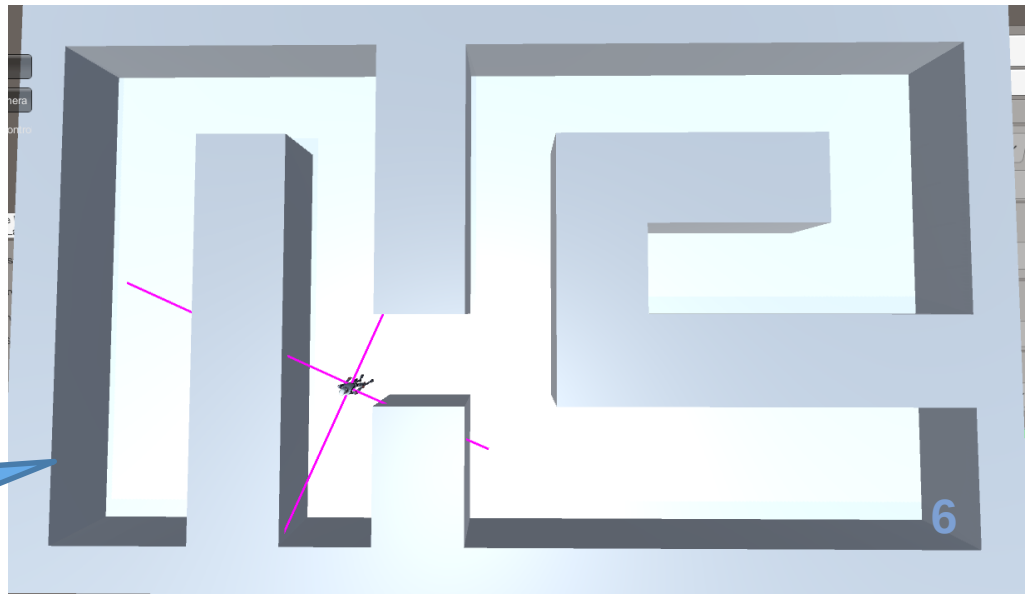
Right Click





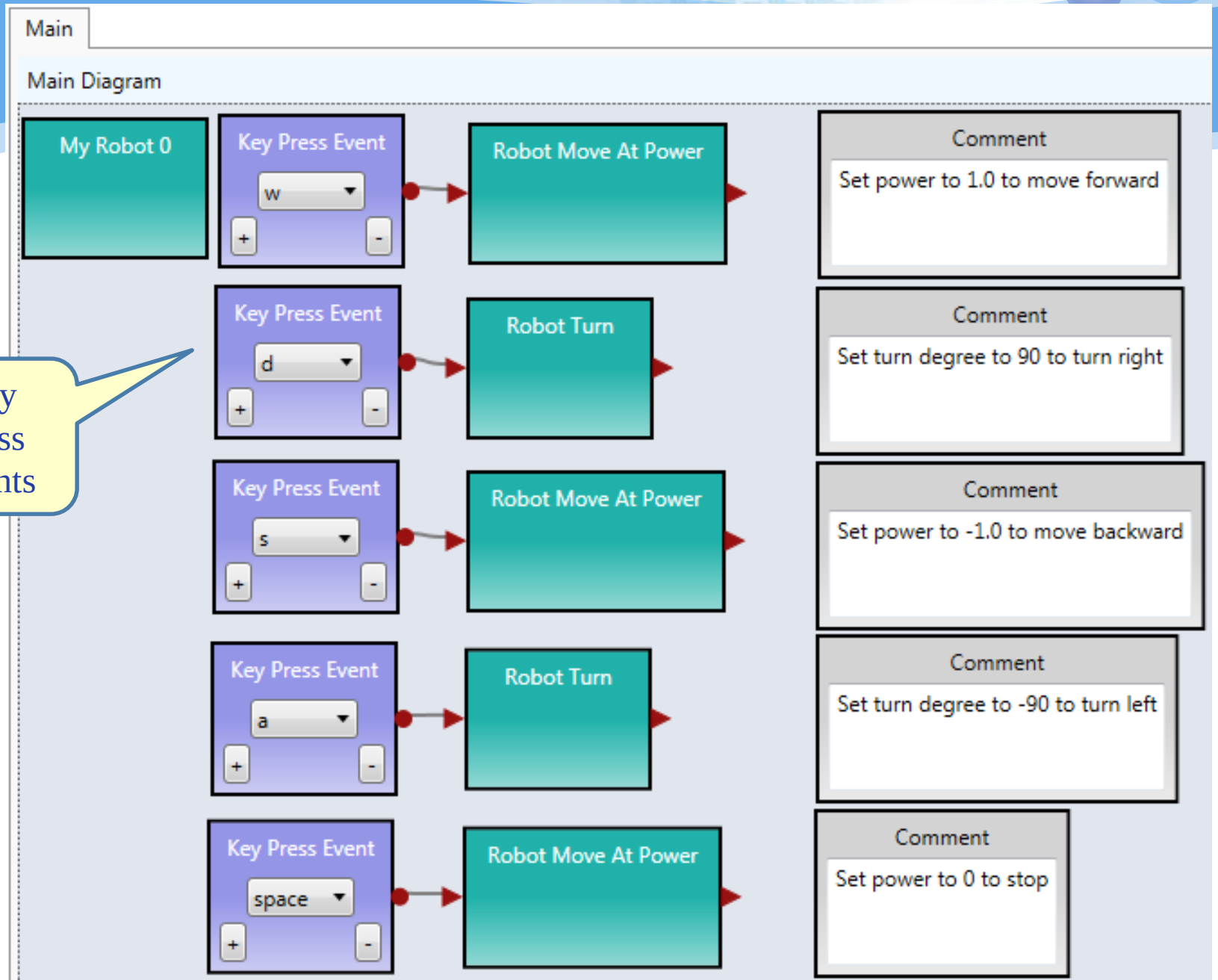






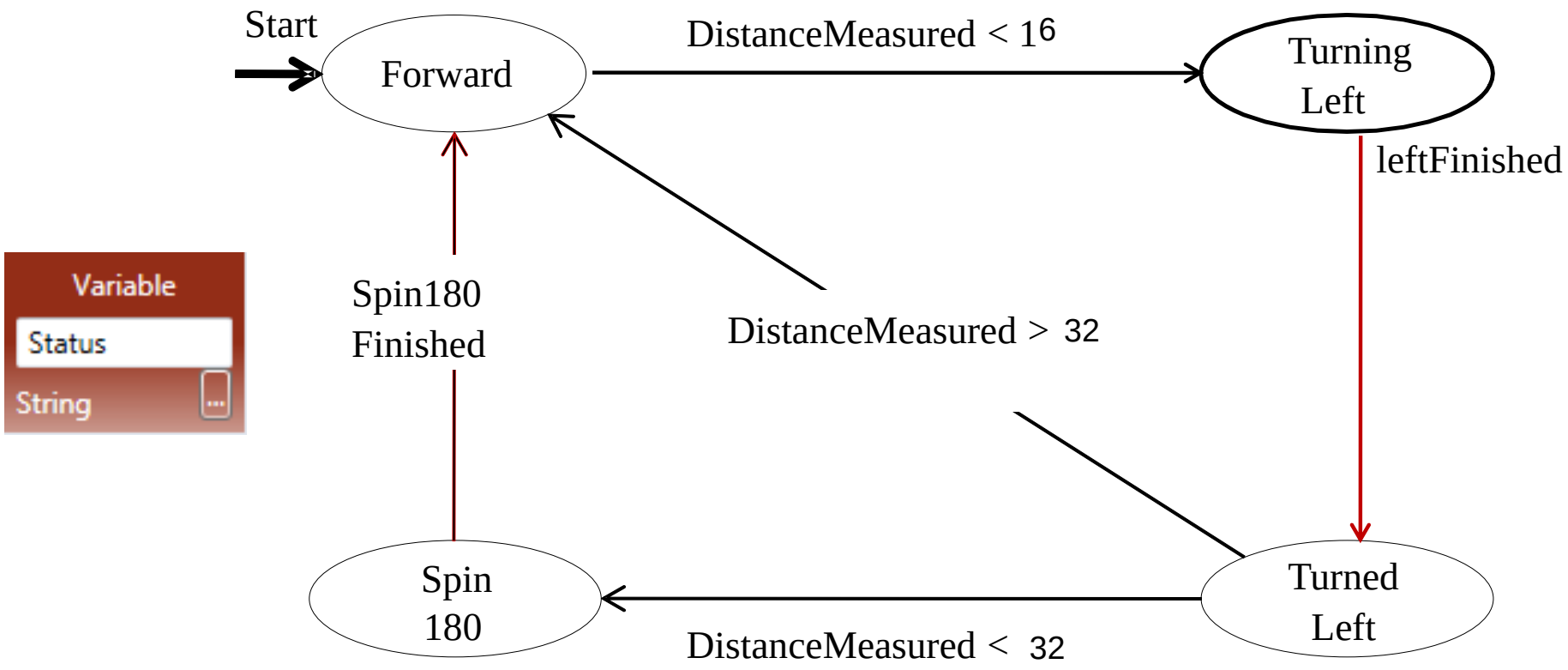
The simulator has two distance sensors, in the front and on the right.

Drive-by-Wire Simulation 遥控仿真



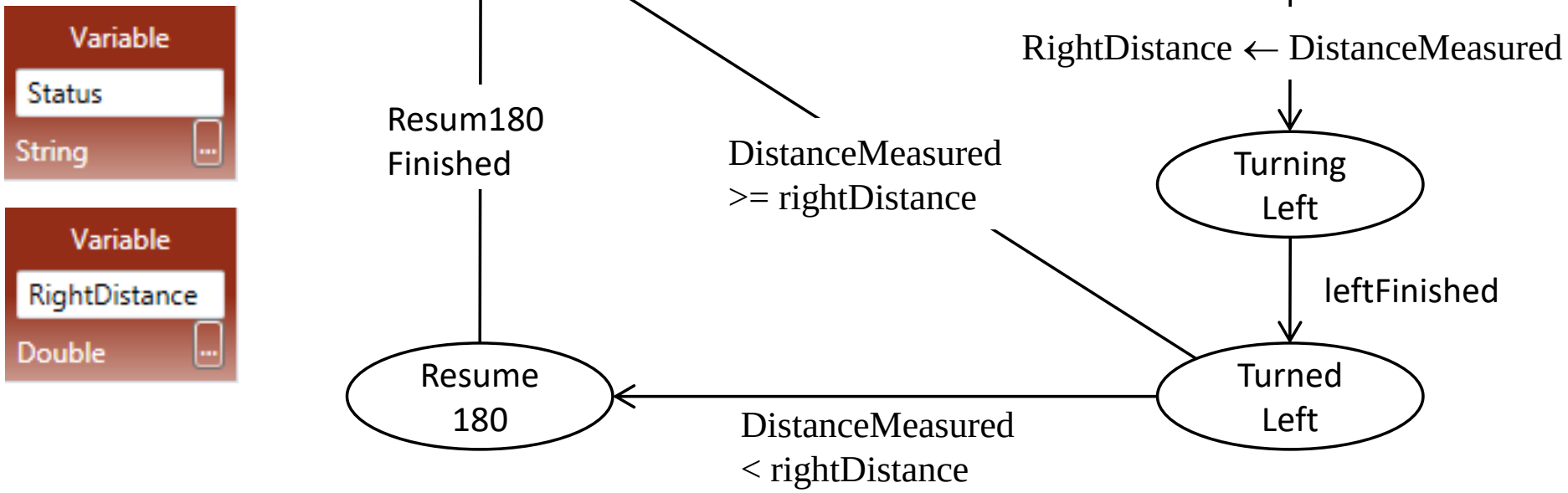
One-Distance Maze Navigation Algorithm
使用一个距离的迷宫航行算法

- 贪婪算法: 使用第一个基于局部信息可行的方法



Two-Distance Maze Navigation Algorithm
使用两个距离的迷宫航行算法

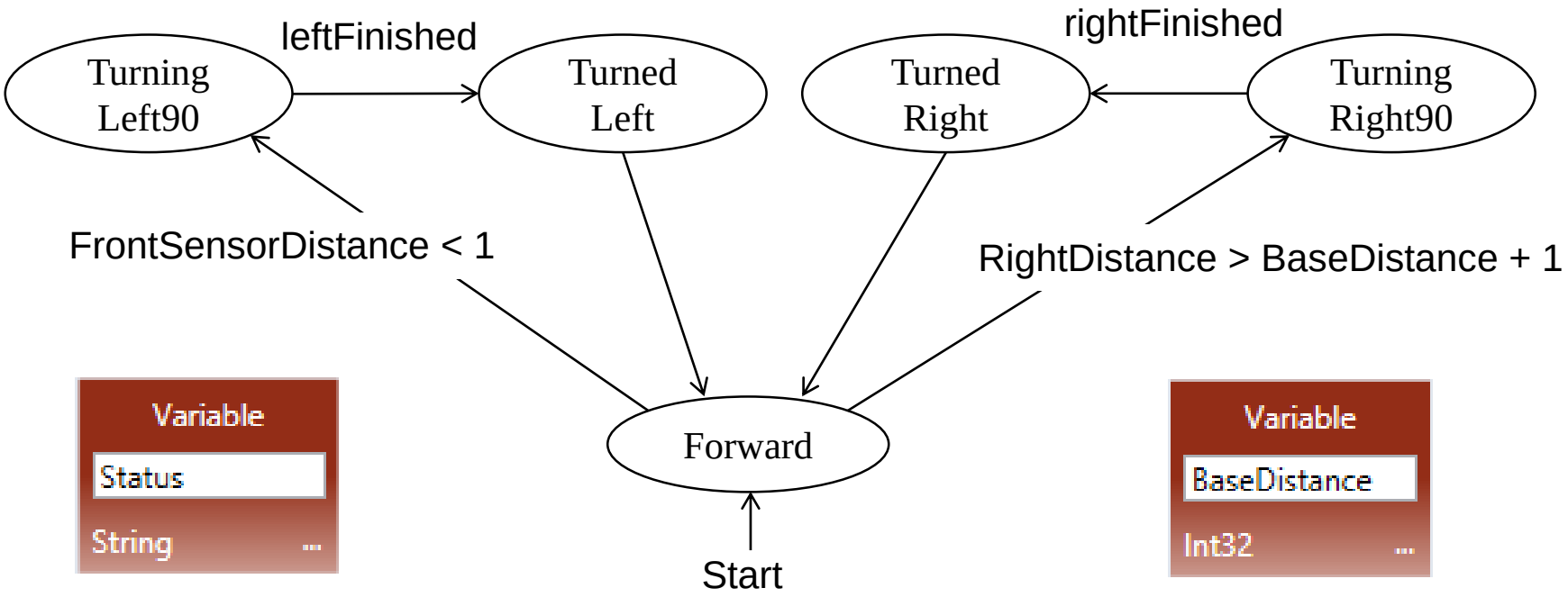
- 局部最优算法: Measure two distances and choose the direction with the longer distance



Wall-Following Maze Navigation Algorithm

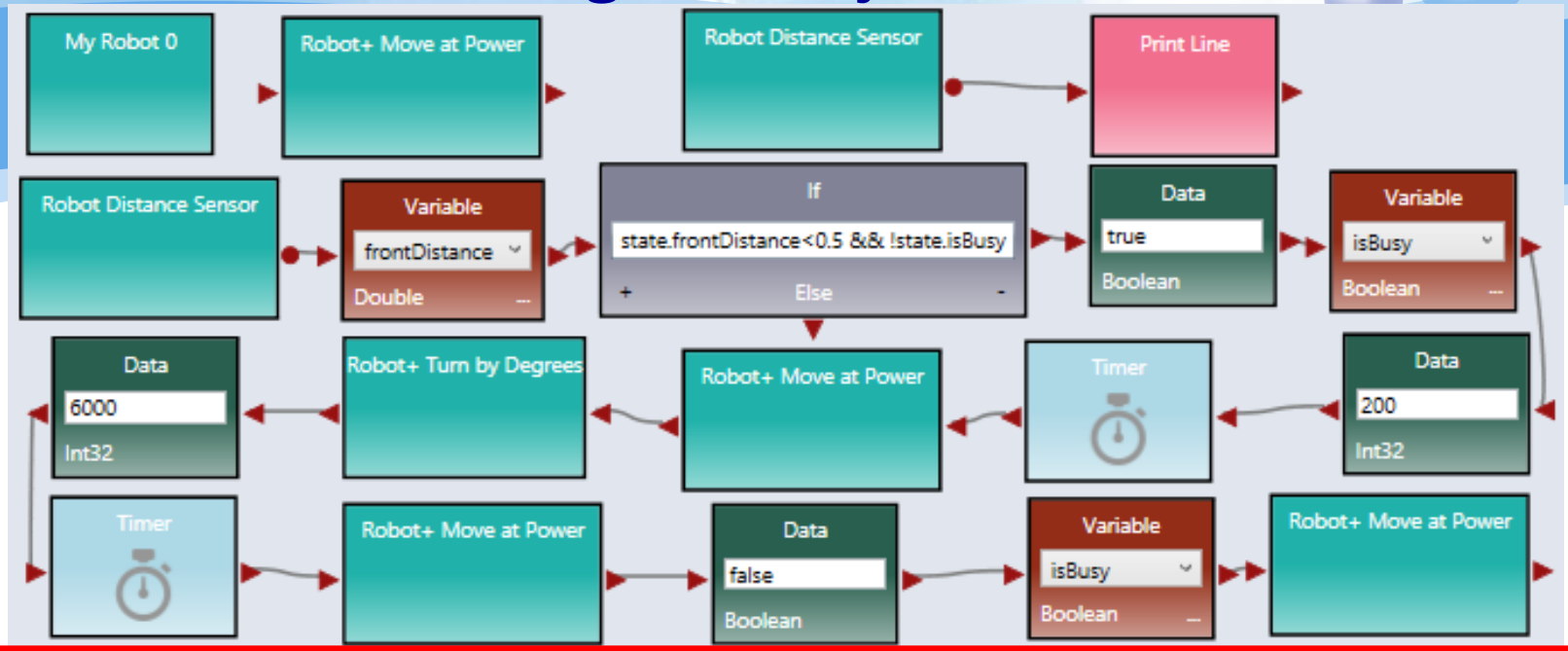
绕右墙算法 Right-Wall-Following:

- Move forward;
- Turn right if the right side is open;
- Turn left if the front is not open.
- It assumes that forwards and turns are accurate.

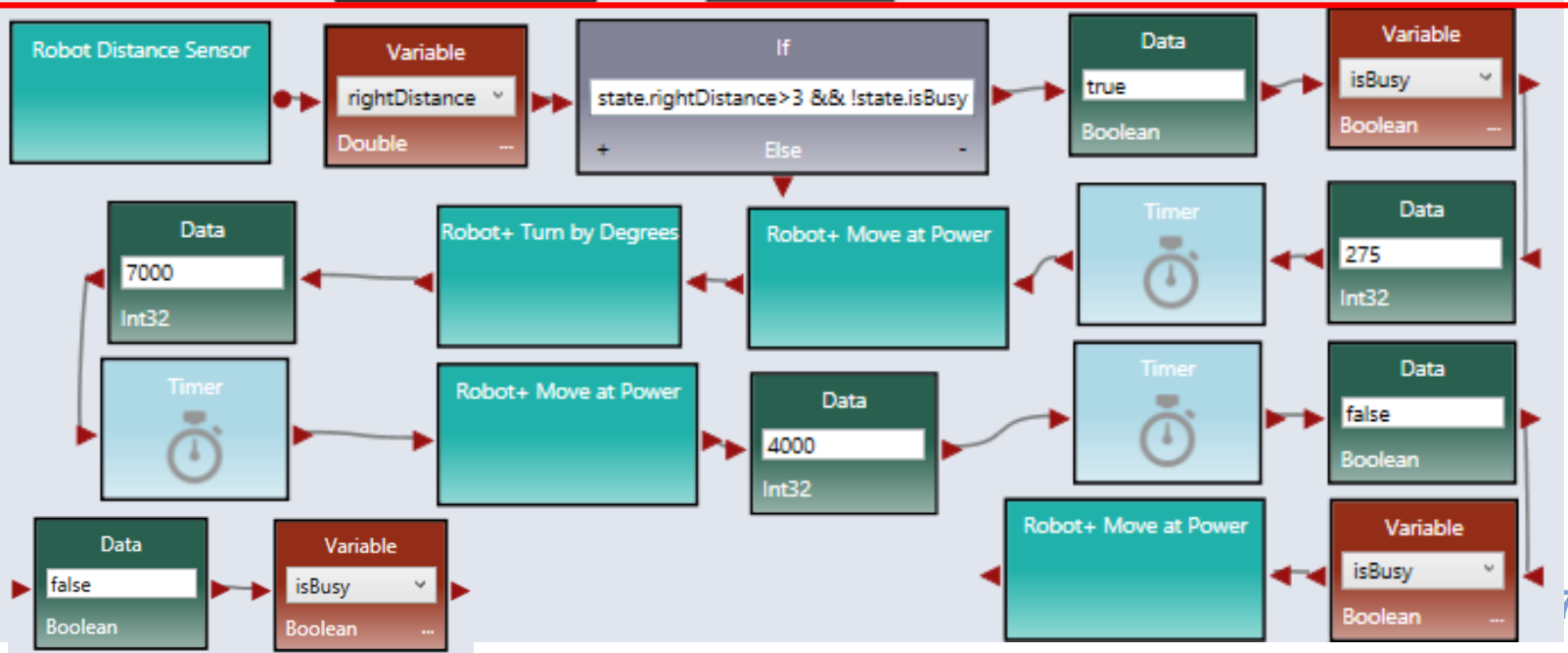


Wall Following in Unity Simulator

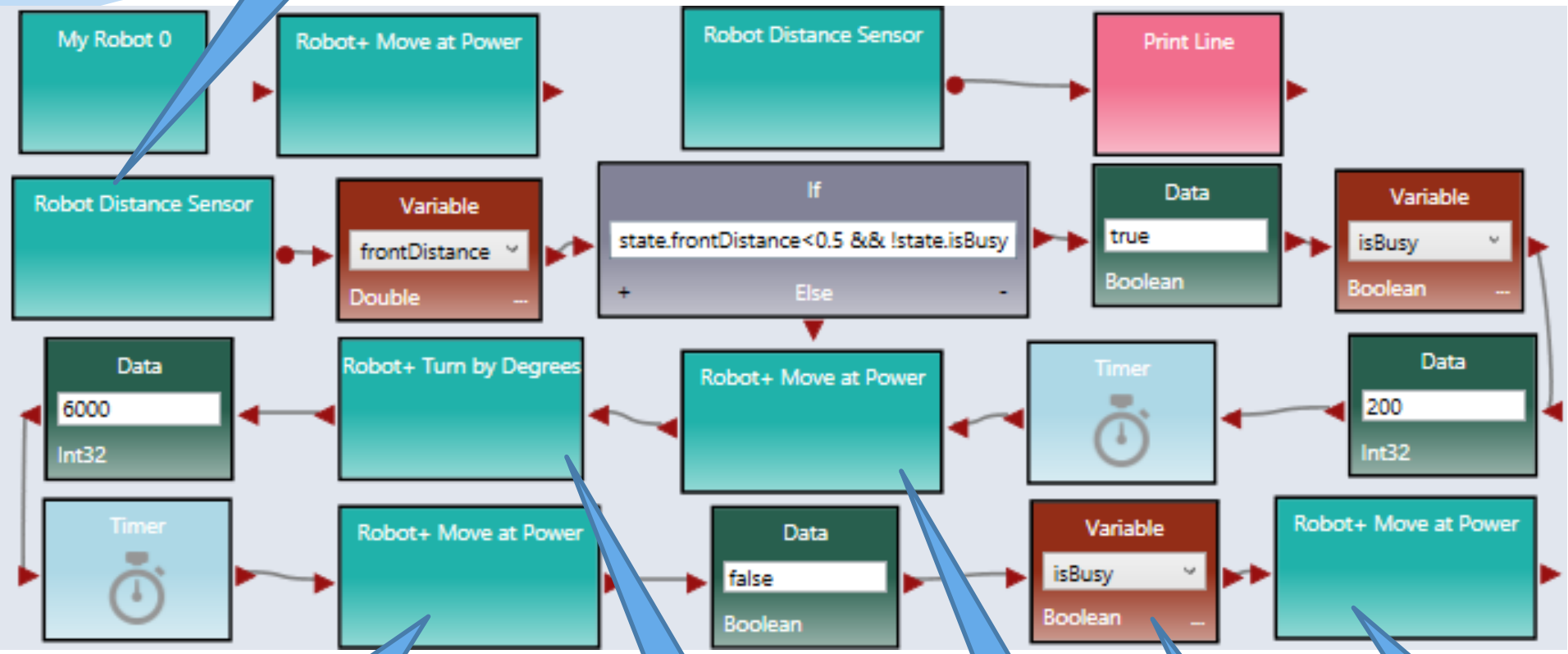
Processing Front Distance Sensor 处理前方距离传感器



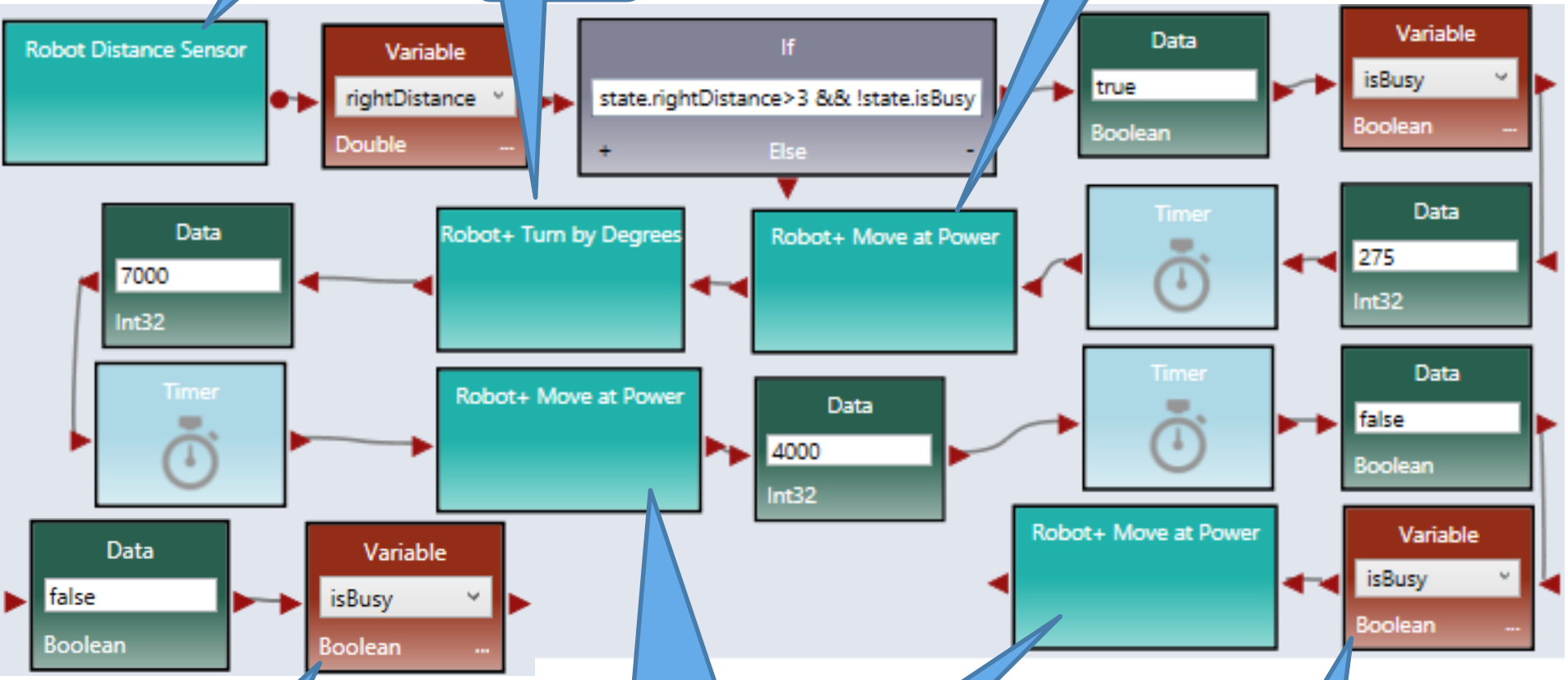
Processing Right Distance Sensor 处理右边距离传感器



Processing Front Distance Sensor
处理前方距离传感器



Processing Right Distance Sensor
处理右边距离传感器



Port= 1

Right 90

Stop

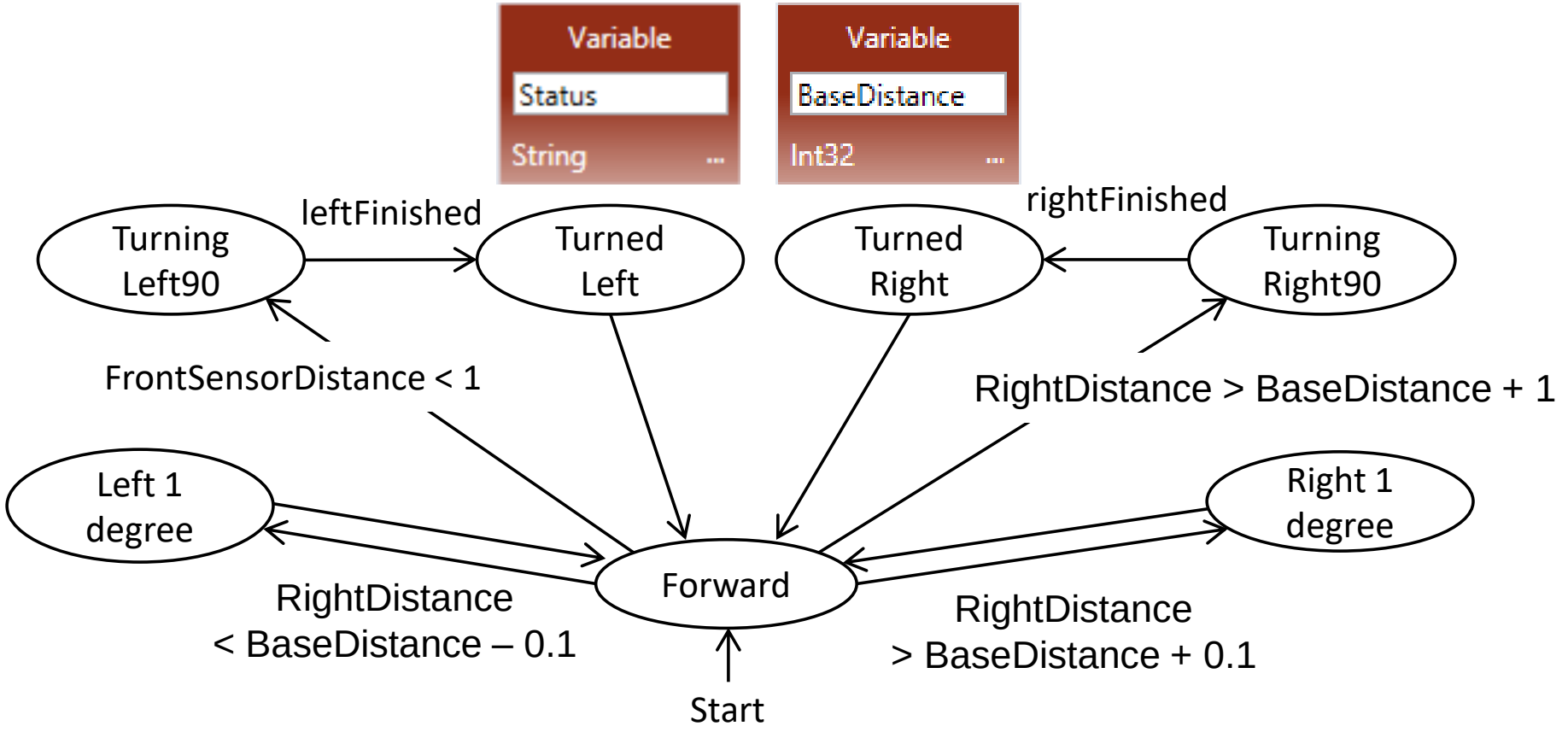
Default

Forward 7 sec to avoid circular move

Forward

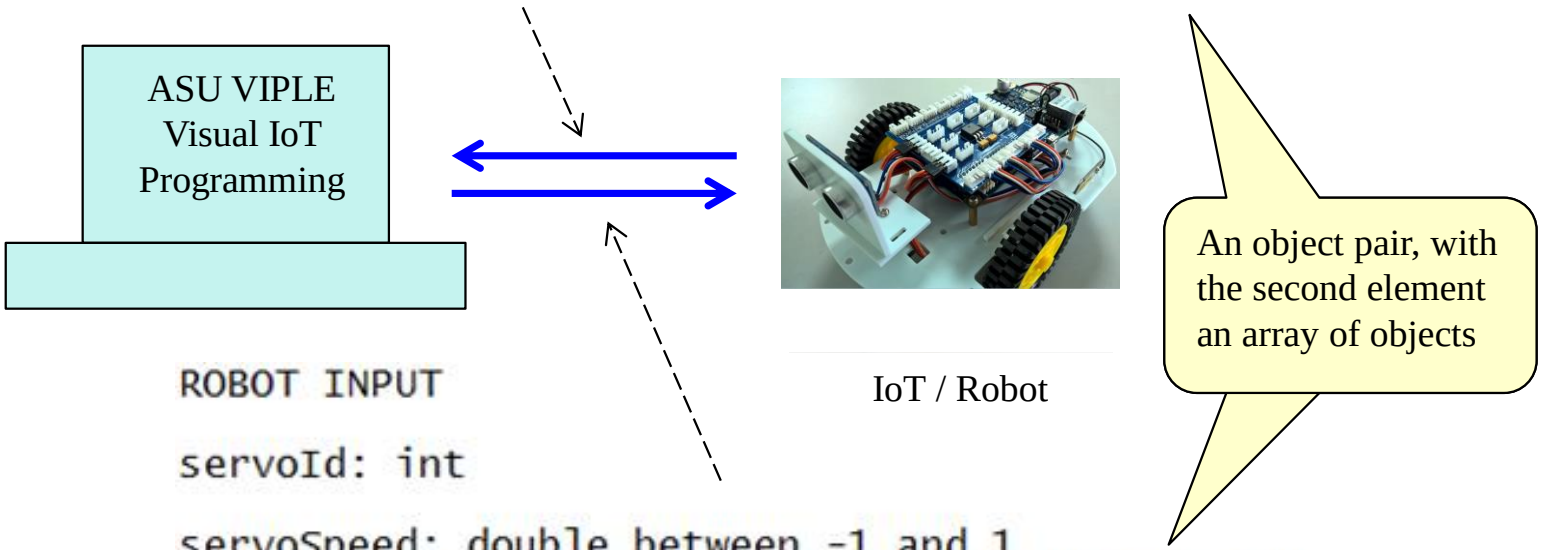
Not busy

自调节算法


Variable
Status
String
Variable
BaseDistance
Int32
Turning Left90
leftFinished
Turned Left
Turned Right
rightFinished
Turning Right90
FrontSensorDistance < 1
RightDistance > BaseDistance + 1
Left 1 degree
Right 1 degree
Forward
RightDistance < BaseDistance – 0.1
RightDistance > BaseDistance + 0.1
Start

Communication between VIPLE & Device: JSON Object

```
ROBOT OUTPUT

name: string (touch, distance, sound, light, color, motorEncoder)
id: int
value: For touch sensor, value will be an int (0 = not pressed and
                                                1 = pressed).
       For other sensors, value will be a double

{"sensors": [{"name":"touch", "id":0, "value":0},
             {"name":"distance", "id":1, "value":12.8}]}
```



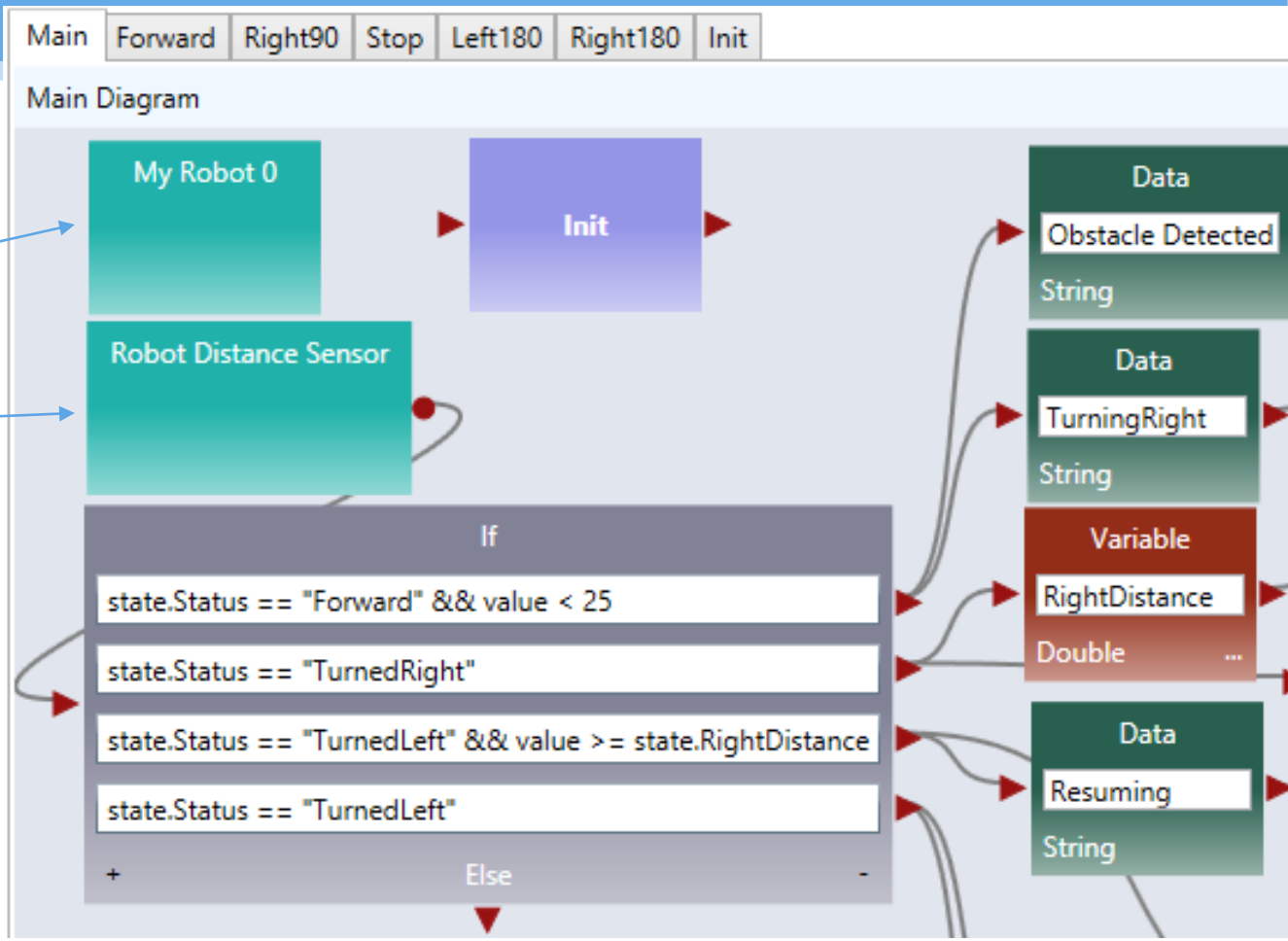

```
ROBOT INPUT

servoId: int

servoSpeed: double between -1 and 1
 - negative values represent a backwards motion

{"servos": [{"servoId":3, "servoSpeed":0.5},
            {"servoId":5, "servoSpeed":-0.5}]}
```

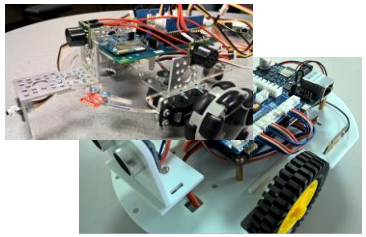
Interfacing with Different Devices
与不同设备的接口

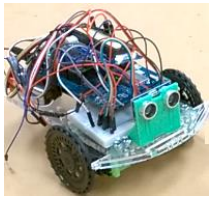
- For open architecture robot, we can program the robot
  对于开放式结构机器人，我们都可以对机器人进行编程:
  - 用VIPLE解释JSON对象
  - 将传感器数据封装到JSON对象中
- For vendor-specific robots, e.g., Lego EV3 and iRobot, 供应商特定的机器人，如乐高EV3和iRobot
  - We cannot program the robots to interface with VIPLE
    我们不能对机器人接口编程与VIPLE通讯
  - We program VIPLE to interface with each type of such robots. I have Lego robots programmed so far.
    我们在VIPLE中编程与这类类型的机器人接口。目前，我们只实在了乐高机器人EV3。

Connecting to Physical Robot: Intel Robot

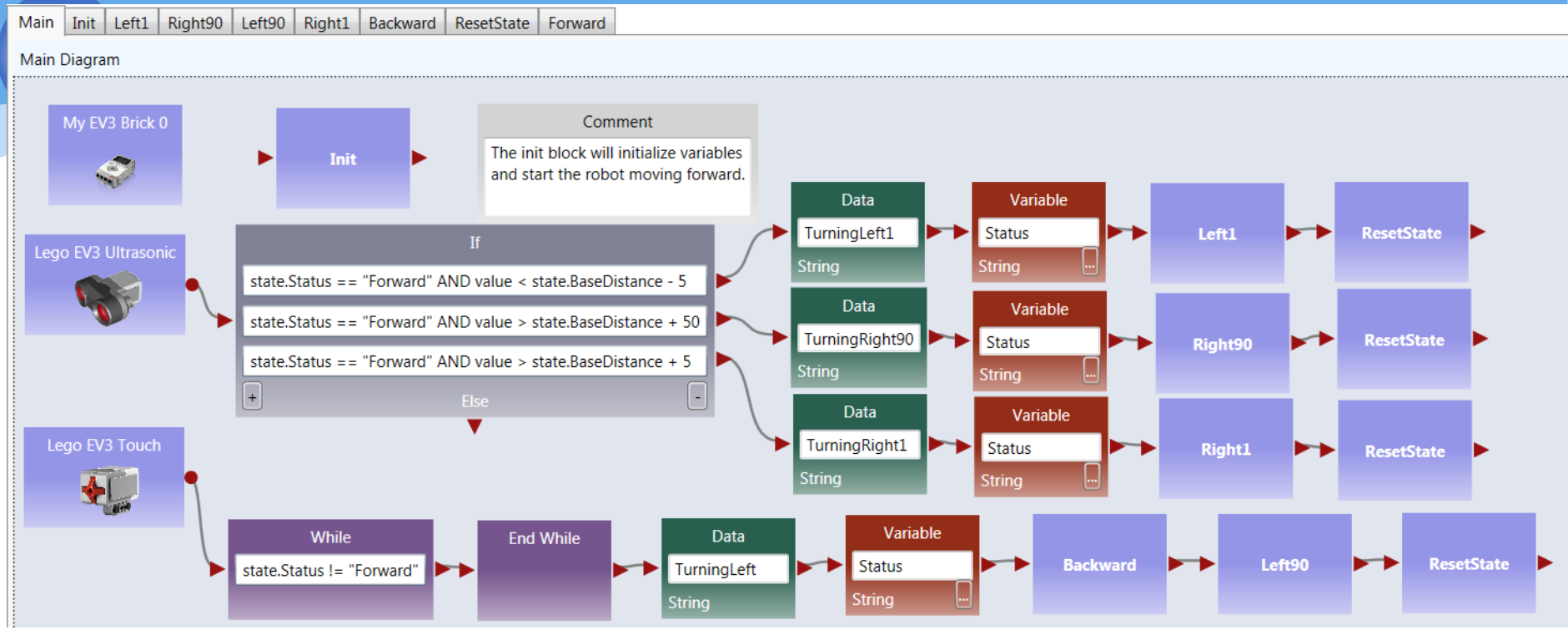
- 乐高EV3释放触摸
- 乐高EV3超声
- 行打印
- 随机
- RESTful服务
- 机器人主机
- 机器人彩色传感器
- 机器人距离传感器
- 机器人驱动器
- 机器人完整协调驱动
- 机器人光传感器
- 机器人运动
- 机器人电机
- 机器人电机编码器
- 机器人声音传感器
- 机器人触觉传感器
- 机器人+移动-动力控制
- 机器人+转动-角度控制
- 简单的对话



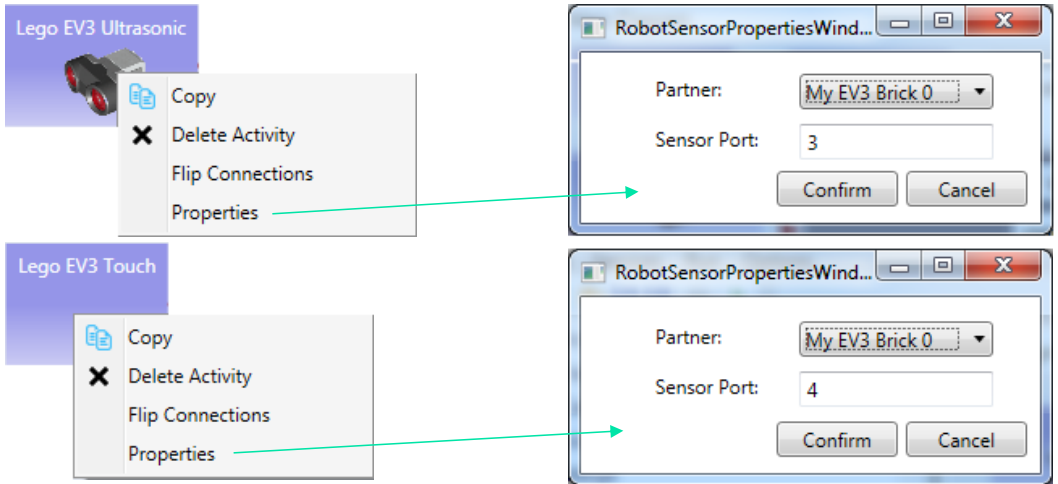




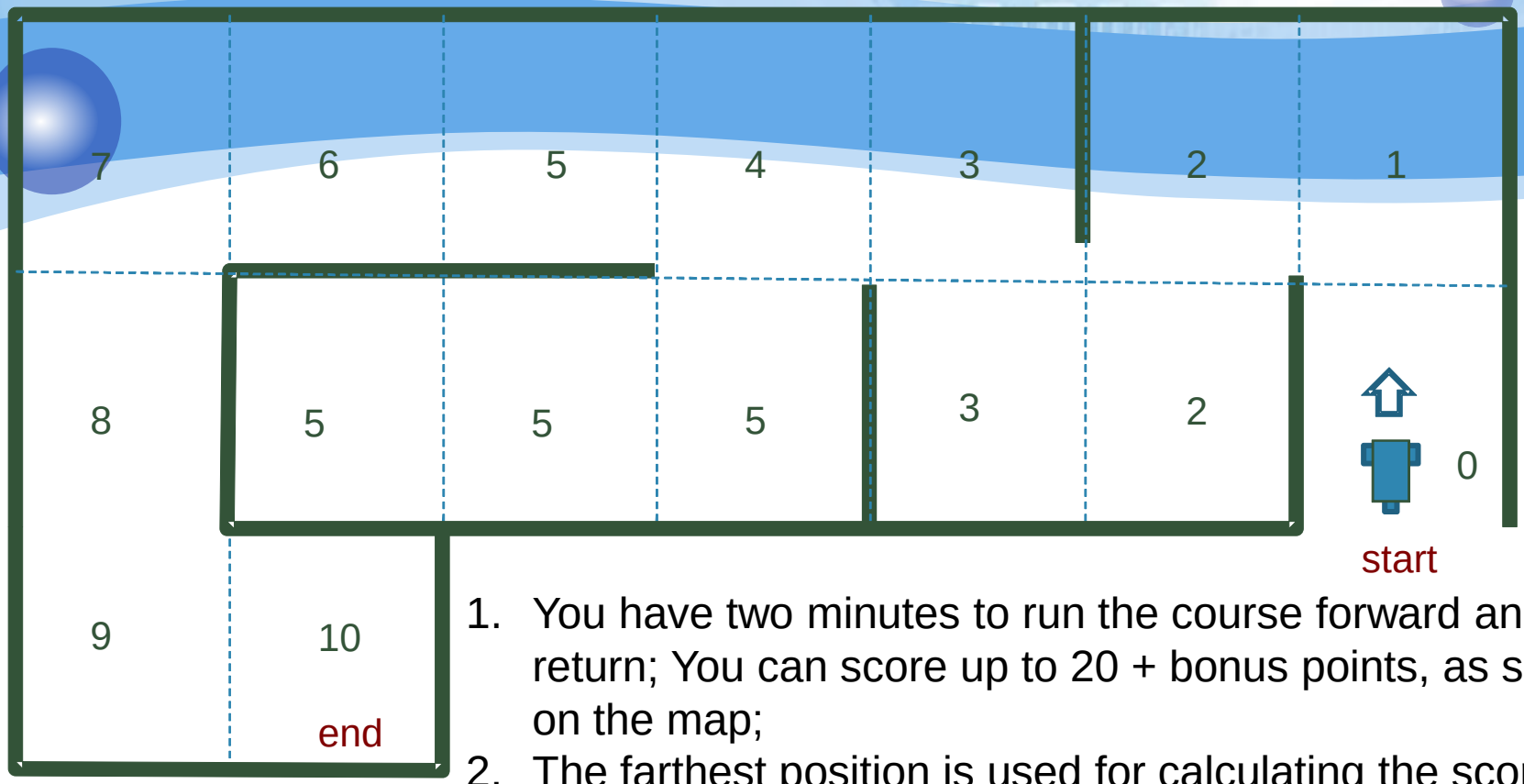
Controlling Physical Robot: Lego EV3 Robot



- Lego EV3 Brick
- Lego EV3 Color
- Lego EV3 Drive
- Lego EV3 Drive for Time
- Lego EV3 Gyro
- Lego EV3 Motor
- Lego EV3 Motor by Degrees
- Lego EV3 Motor for Time
- Lego EV3 Touch Pressed
- Lego EV3 Touch Released
- Lego EV3 Ultrasonic



Maze Navigation Game with Artificial Intelligence



1. You have two minutes to run the course forward and return; You can score up to 20 + bonus points, as shown on the map;
2. The farthest position is used for calculating the score;
3. If forwarding failed in the middle, you can take the robot to the end position to run the backward part;

Grading Scales:

4. If you use sensor to detect the front wall, + 10% bonus
5. If you use sensor(s) to detect front and side walls + 20% bonus points;
6. If you do not touch robot for the return trip, you receive 2 bonus points.