PRESENTATION 5 · LEARNING WITH C

Hard maze

A bigger maze, narrower corridors, more dead ends, and the same four branches.

TheBotCup 2026 · AiBotCup Simulator



REMEMBER FROM PRESENTATION 4

The right-hand wall-follower

```
// Session 5 · Hard maze

void Game1() {
    if (US_Front <= 20) {                    // Ahead
        WheelLeft = -2; WheelRight = 2;
    } else if (US_Right <= 22) {             // TooClose
        WheelLeft = 2.5; WheelRight = 4.5;
    } else if (US_Right >= 45) {             // Lost
        WheelLeft = 4.5; WheelRight = 1.5;
    } else {                                 // Forward
        WheelLeft = 4.5; WheelRight = 4.5;
    }
}
```

Ahead wall in front → spin left

TooClose right wall near → curve left

Lost right wall gone → curve right

Forward the else → straight

PART 1

The mission: Session 5

Solve the big maze

SESSION 5 · HARD MAZE

START to FINISH, the long way

5

Session 5 · Hard maze

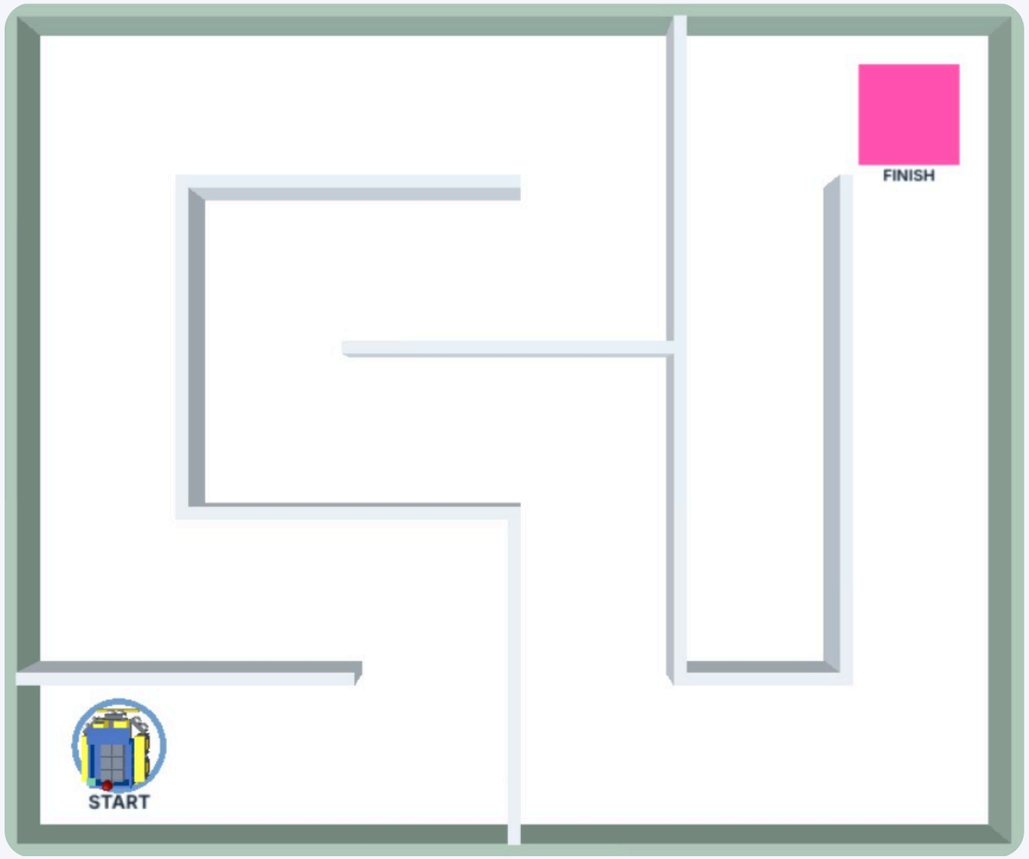
Solve the big maze: from START (bottom left) to the pink FINISH pad (top right).

Reach FINISH: +300. Time limit 8:00.

Not done yet

START bottom left, facing up, a wall just ahead.

FINISH top right: **+300**. Time limit **8:00**.



WHAT IS HARDER?

Session 4 and Session 5 side by side

	Session 4 · Simple	Session 5 · Hard
Field	240 × 180 cm	300 × 250 cm
Cells	4 × 3 = 12	6 × 5 = 30
Corridor width	60 cm	**50 cm** (robot is 22 cm)
Dead ends	few	many
Points · time limit	150 · 5:00	300 · 8:00

PREDICT BEFORE YOU TEST

Will the same program still work?

60 cm corridor

In the middle, US_Right reads about **31**.

50 cm corridor

In the middle, it reads about **24**.

Good band 23–44

24 is still inside, but only **just**.

Vote: **yes**, **no**, or **yes but slower**? Then we test.

PART 2

Step by step in C

Open · check · build · run · understand

STEP 1 · OPEN

Reuse your program from last time

1 Sessions → **Session 5** → **2** **AI** → **Open from PC** → Maze Runner → **3** Check the 4 branches → **4** **Build** → OK

No file? Make it again from the code on the **recap** slide: 4 branches, 2 minutes.

Have the robot file (**.tbcbin**)? Skip to Load → **Load robot file from PC**.

STEP 2 · CHECK

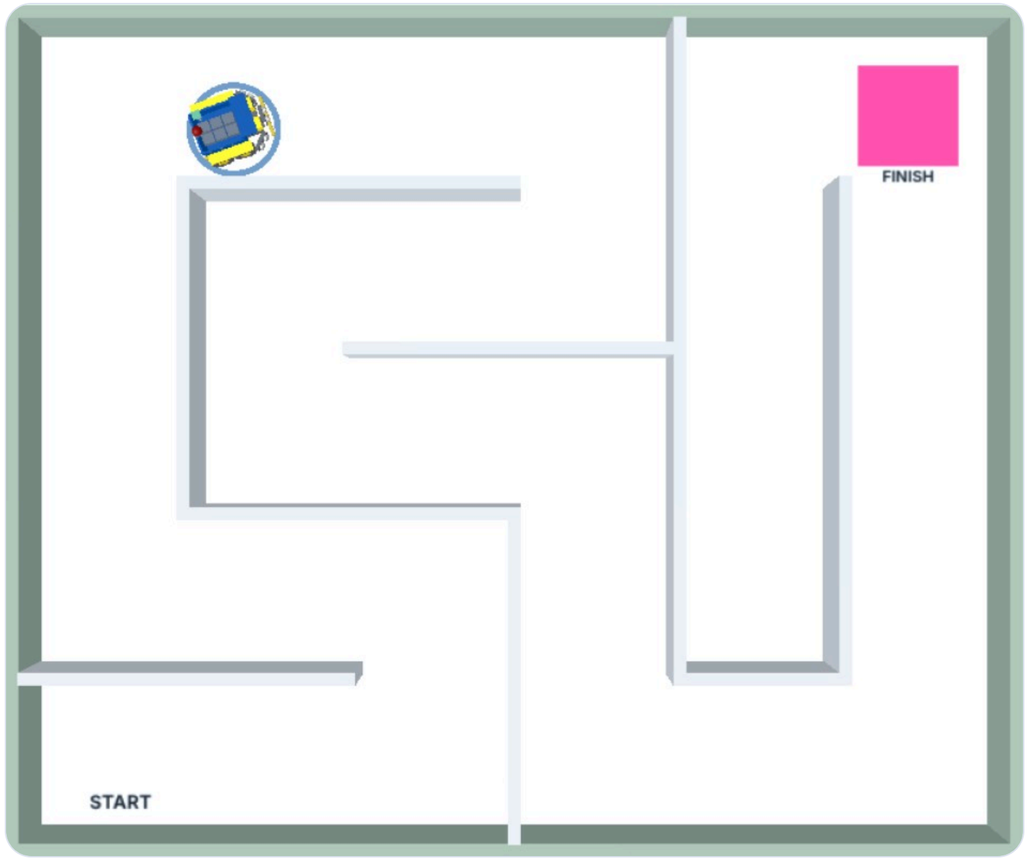
Nothing changes: the same four branches

```
// Session 5 · Hard maze

void Game1() {
    if (US_Front <= 20) {                    // Ahead
        WheelLeft = -2; WheelRight = 2;
    } else if (US_Right <= 22) {             // TooClose
        WheelLeft = 2.5; WheelRight = 4.5;
    } else if (US_Right >= 45) {             // Lost
        WheelLeft = 4.5; WheelRight = 1.5;
    } else {                                 // Forward
        WheelLeft = 4.5; WheelRight = 4.5;
    }
}
```

No Time, no Position, no map of the maze.

The robot only **reacts** to what its ultrasonic sensors hear.



STEP 3 · RUN

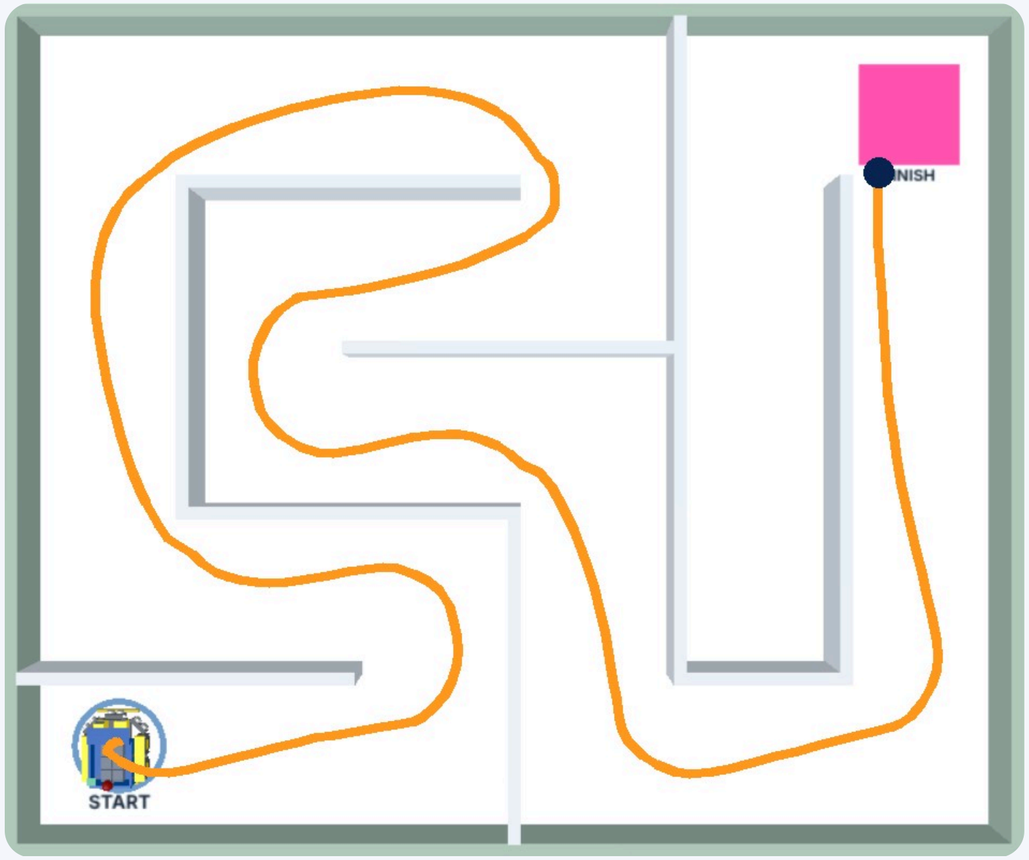
Load on BLUE, Play, watch

Ultrasonic Sensors

Front	**74**	(US_Front)
Left	**23**	(US_Left)
Right	**38**	(US_Right)

At 20 s: Front **74**, Right **38** → no branch above is true, so **Forward**.

Use the **Top** camera: the wall stays on the robot's right.



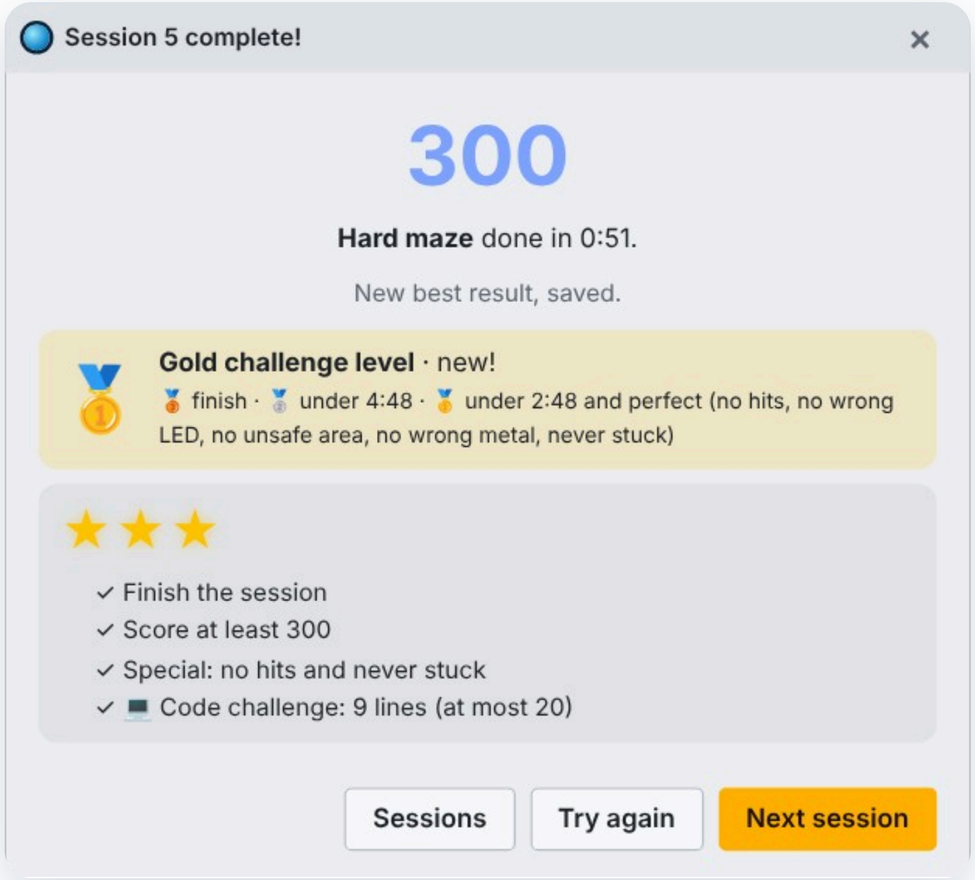
THE PATH · RECORDED

The whole trip in one line

A long winding way, but **never lost**.

Trace it with your finger: the walls are always on the right of the line.

The answer to our question: **yes, it works**.



RESULT · TESTED

300 points in 0:51 · Gold

Medals · Bronze: finish · Silver: under 4:48 · Gold: under 2:48 and perfect.

Stars · finish · score 300 · no hits and never stuck.

Code challenge · at most 20 lines. Still 9.

UNDERSTAND IT

How four branches handle every place

Corridor

Forward, with small TooClose corrections.

Corner

Wall ahead → **Ahead** spins left until the way is clear.

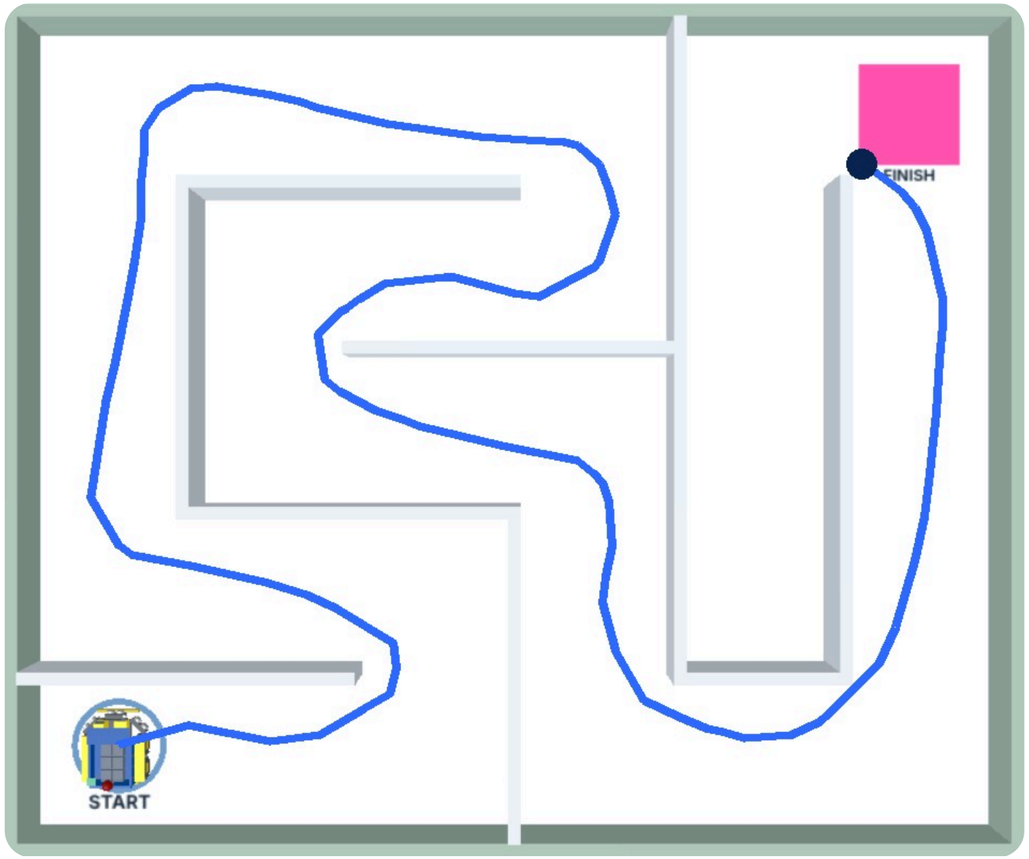
Opening

Right wall ends → **Lost** curves right into it.

Dead end

Ahead again and again: the robot turns all the way round and drives out.

No branch says “dead end”: the behaviour **emerges** from simple rules.



LEFT OR RIGHT? · TESTED

Here the left hand is faster

Maze	Right	Left
Session 4	**0:25**	0:31
Session 5	0:51	**0:48**

Both always arrive. Which is faster depends on the maze.

WHY DOES IT ALWAYS WORK HERE?

All the walls are joined to the outside wall.

Works

START and FINISH are at the edge, and every wall touches the outside: following one wall visits them all.

Can fail

If the goal sat next to a wall “island” in the middle, the robot could circle the outside forever.

DID NOT WORK?

Check first, then tune carefully

What you see	What to do
Program behaves differently from Session 4	Compare with the recap slide: one number may be wrong
Old program still on the robot	Build again, then Load on BLUE
Touches walls in narrow corridors	TooClose **<= 22**, Lost **>= 45**: change one number only
Wanders back towards START	That is normal: the rule visits dead ends; let it run
Time limit too close	Try the left-hand version (0:48 in our test)

NEXT TIME

Presentation 6 · Iron only

The robot starts **mining**: light and colour sensors find the metals on the floor.