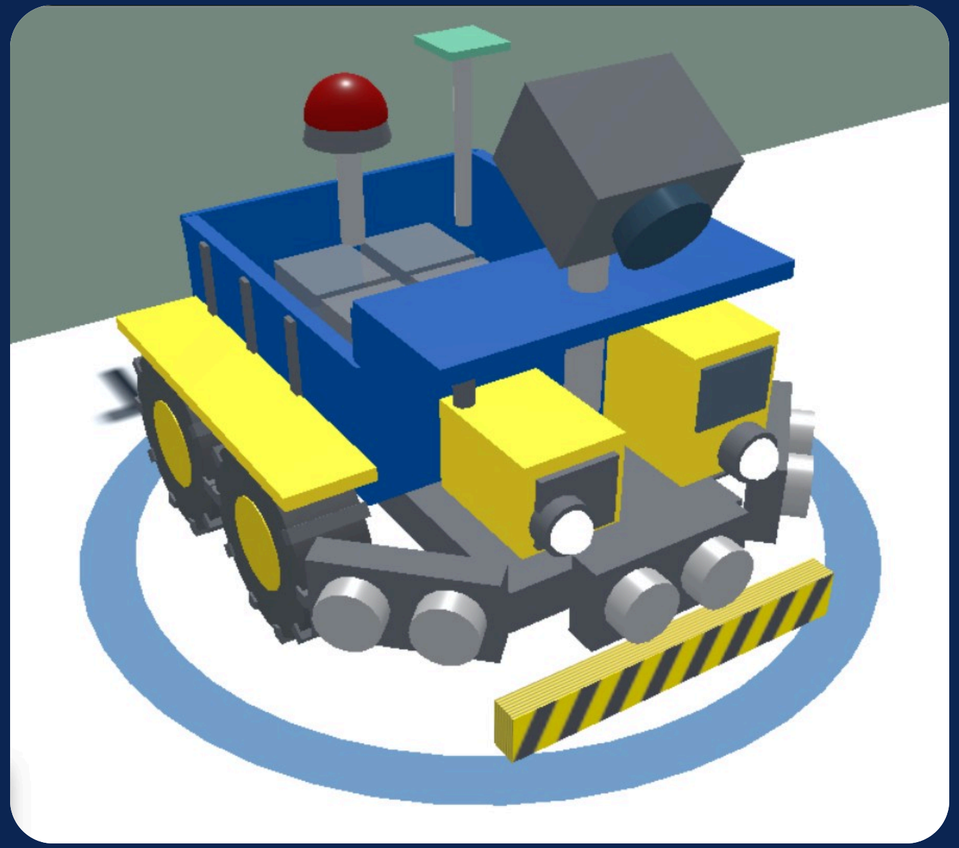
UNIVERSITY · PRESENTATION 7 · LEARNING WITH PYTHON

Avoid by camera

Five unsafe areas between START and FINISH: see their red warning bands and steer around them.

TheBotCup 2026 · AiBotCup Simulator · Advanced



RECAP

From a good place to a bad one

Session 16

Line up and collect

Steer by the column of a target.

Session 17 · last time

Pond by camera

A colour mask finds the water; count the bottom rows.

Session 18 · today

Avoid by camera

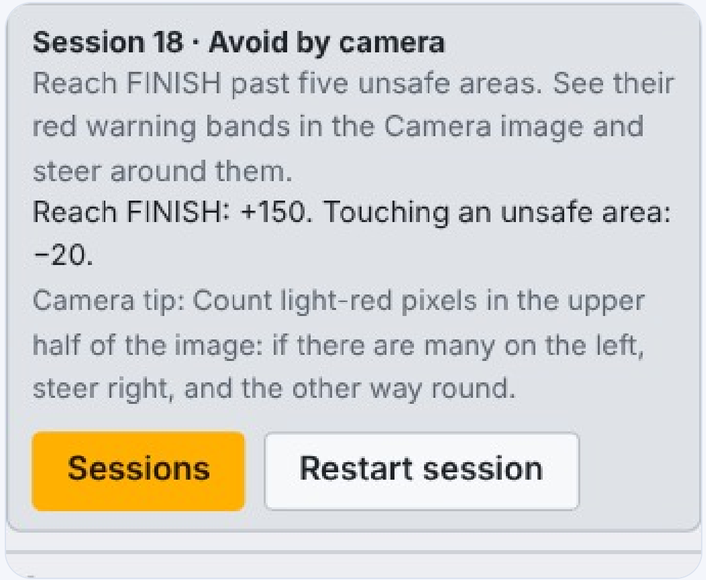
Count red pixels on each side and steer away.

New today: the robot does not drive **to** what it sees, it drives **away** from it, and uses its position to keep heading for FINISH.

PART 1

The mission: Session 18

Reach FINISH past five unsafe areas



SESSION 18 · AVOID BY CAMERA

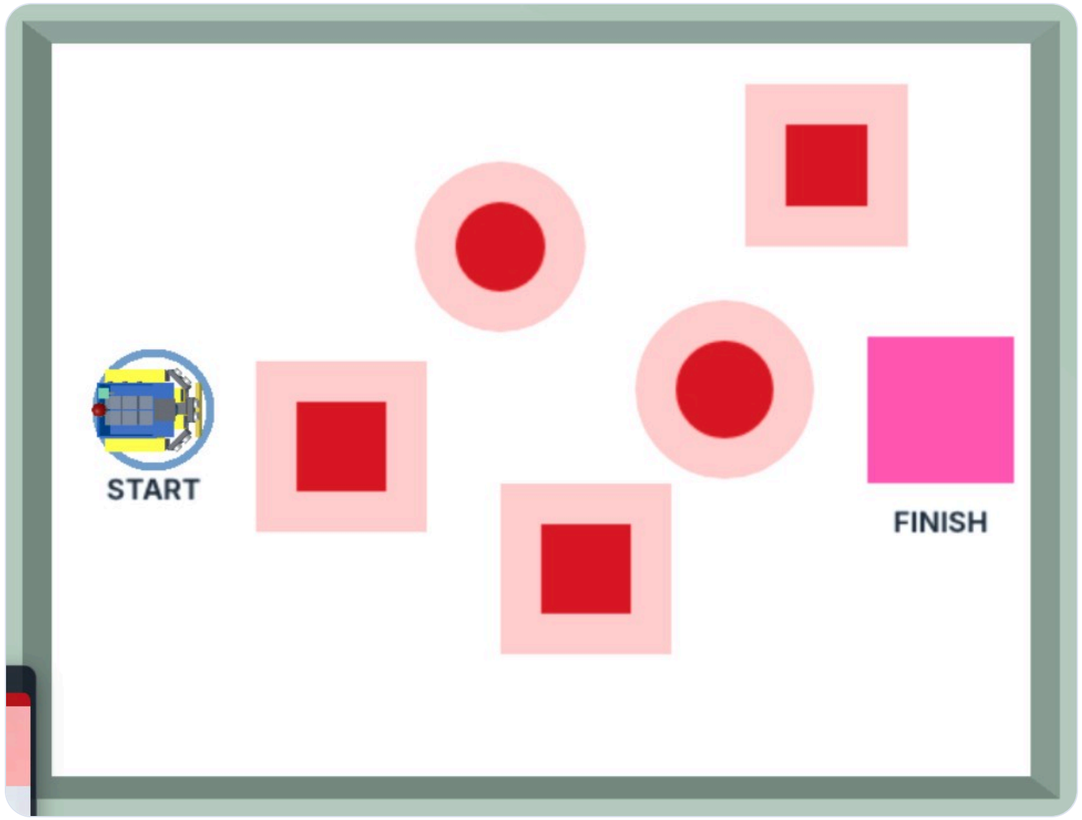
Get there without touching

Reach the pink FINISH: **+150**.

Touching an unsafe area (dark red): **−20** each time.

Time limit **4:00** · code challenge only **20 lines**.

The light-red **warning band** is safe to cross. It is your early warning.



THE FIELD

Five dangers in the way

START (25, 90), facing east (Compass 270).

FINISH x 200–236, y 72–108: centre (218, 90).

5 unsafe areas, 3 squares and 2 circles, each in a warning band about 10 cm wide.

A straight line from START hits the first one.

PART 2

Seeing danger

Red on the left, red on the right

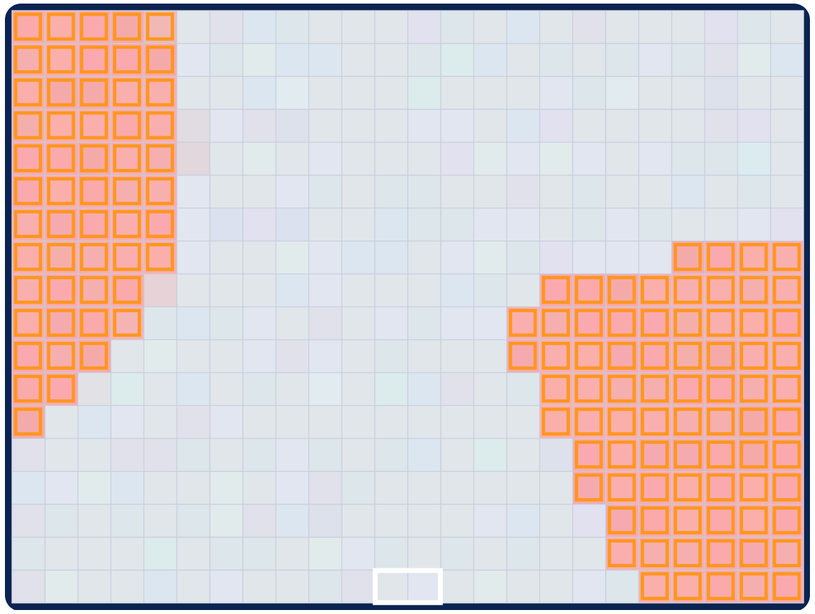
WHAT THE CAMERA READS

Three new colours

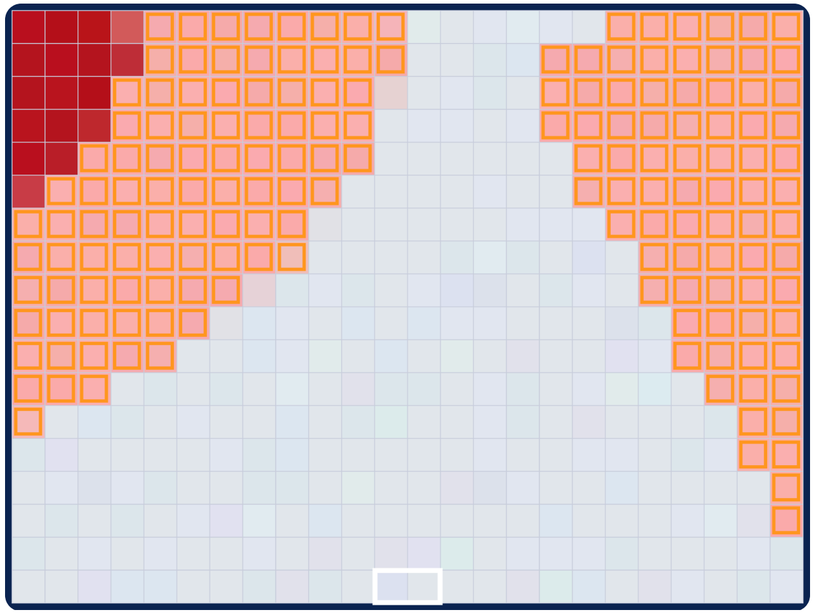
Surface	R, G, B (measured)	Test
Warning band	247, 178, 174	**R > 230 · G 145–205 · B 145–205**
Unsafe area	183, 20, 30	deep red: never reach it
White floor	226, 229, 239	all three high
FINISH	pink: lots of red and blue	reached by position

TWO REAL IMAGES · THE WARNING MASK

Count the red on each side



3.5 s: left **54**, right **77** → turn **left**.



6.8 s: left **90**, right **77** → turn **right**.

Split the image at column CAM_W / 2 = 12. More red on one side? Turn to the **other** side.

DO NOT LOSE THE WAY

Aim at FINISH with position and compass

goal = atan2d(90 − PositionY, 218 − PositionX) − 90
The compass heading that points at the FINISH centre (218, 90).

turn = angleDiff(goal, Compass) × 0.05
How far we are off, in degrees, made into a wheel difference. turn > 0: turn left.

At 9.5 s	Position	Compass	goal	off by	turn
real values	(159, 66)	247°	292°	45°	2.25

Then **WheelLeft = 3 − turn** and **WheelRight = 3 + turn**: the robot curves back towards FINISH.

PART 3

Step by step in Python

Two ideas in 20 lines

THE PLAN

Aim at FINISH, unless red says no

1. Scan the image: count warning pixels in the **left** and **right** half.
2. Compute **turn** from the heading to FINISH.
3. More than 3 red pixels in view → **override**: turn = −2.5 if left is redder, else +2.5.
4. Drive: **WheelLeft = 3 − turn**, **WheelRight = 3 + turn**.

No if chain of five branches this time: one value, turn, decided in two steps.

STEP 1 · LINES 3–12

Count red on each side

```
3  def game1():
4      redL = 0; redR = 0
5      for y in range(CAM_H):
6          for x in range(CAM_W):
7              p = Camera[y][x]
8              if p[0] > 230 and 145 < p[1] < 205 and 145 < p[2] < 205:   # warning band
9                  if x < CAM_W // 2:
10                     redL += 1
11                 else:
12                     redR += 1
```

redL, redR: warning pixels left and right of column 12.

One test with three parts recognises the pale red band.

STEP 2 · LINES 13–18

Aim, override, drive

```
13      goal = atan2d(90 - PositionY, 218 - PositionX) - 90             # heading to FINISH
14      turn = angleDiff(goal, Compass) * 0.05                          # > 0: turn left
15      if redL + redR > 3:                       # band in view: away from the redder side
16          turn = -2.5 if redL > redR else 2.5
17      WheelLeft = 3 - turn
18      WheelRight = 3 + turn
```

goal / turn: heading to (218, 90).

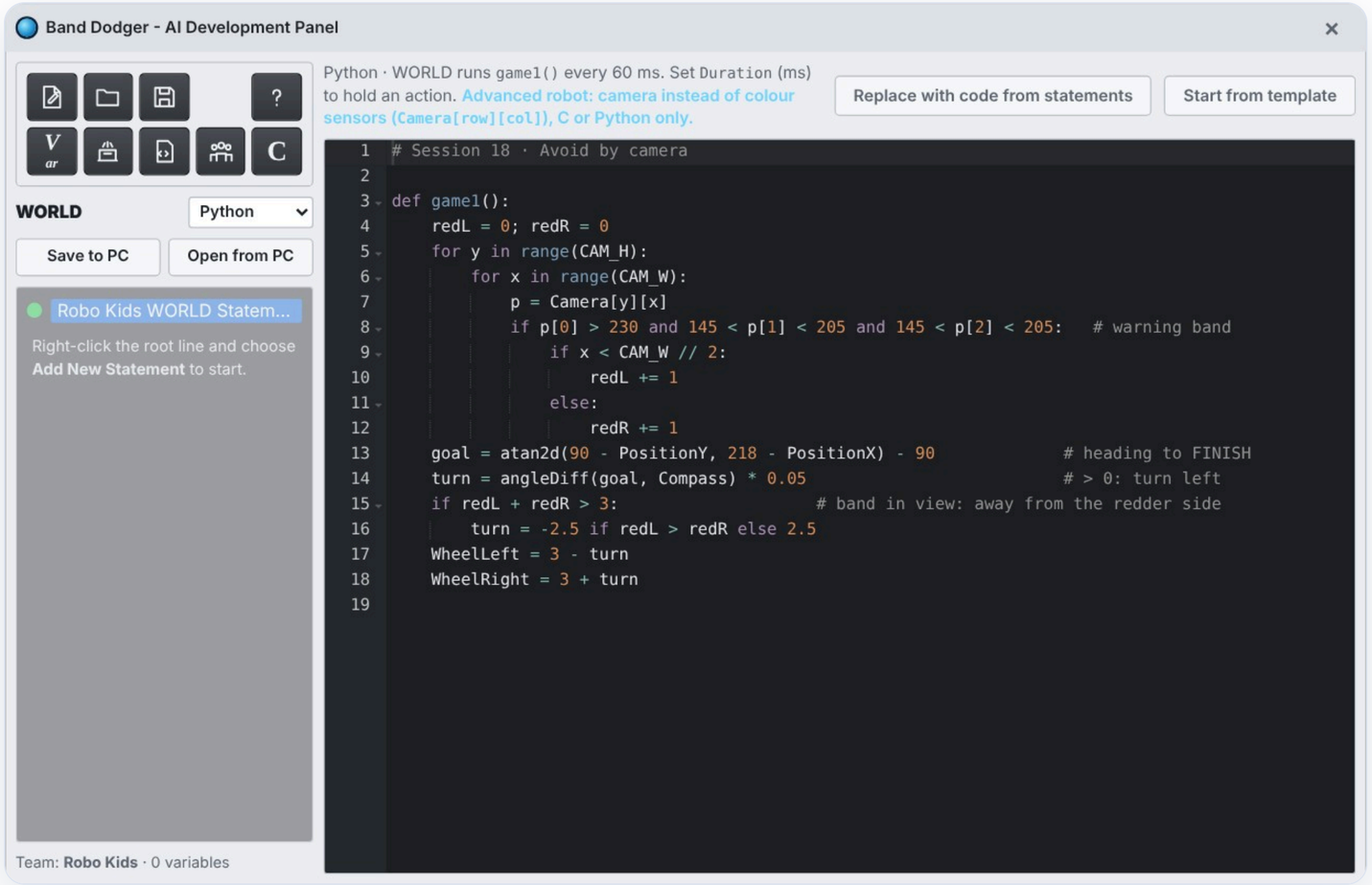
> 3 red: steer away from the redder side.

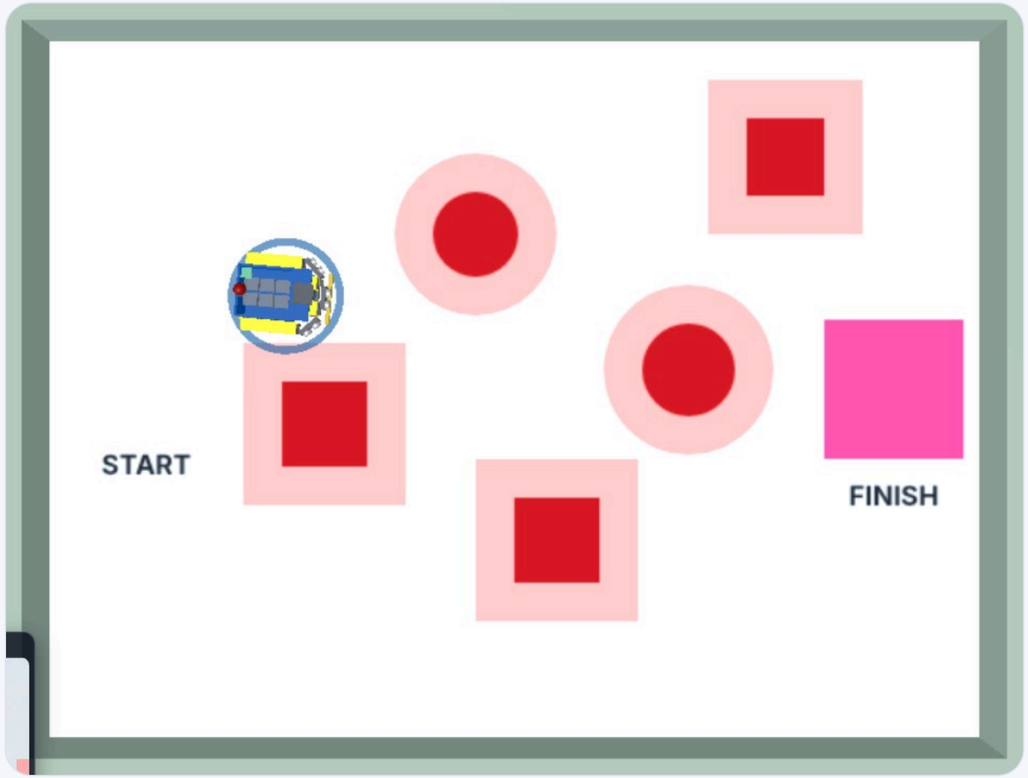
3 ∓ turn: one formula for every case.

BUILD AND LOAD

The whole program, built

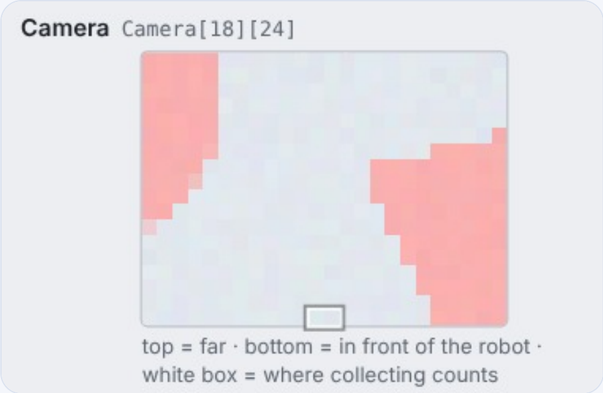
1. WORLD menu: **Python**.
2. Type the program, name it **Band Dodger**.
3. Press **Build**, then **OK**.
4. Press **Load** on the BLUE side.





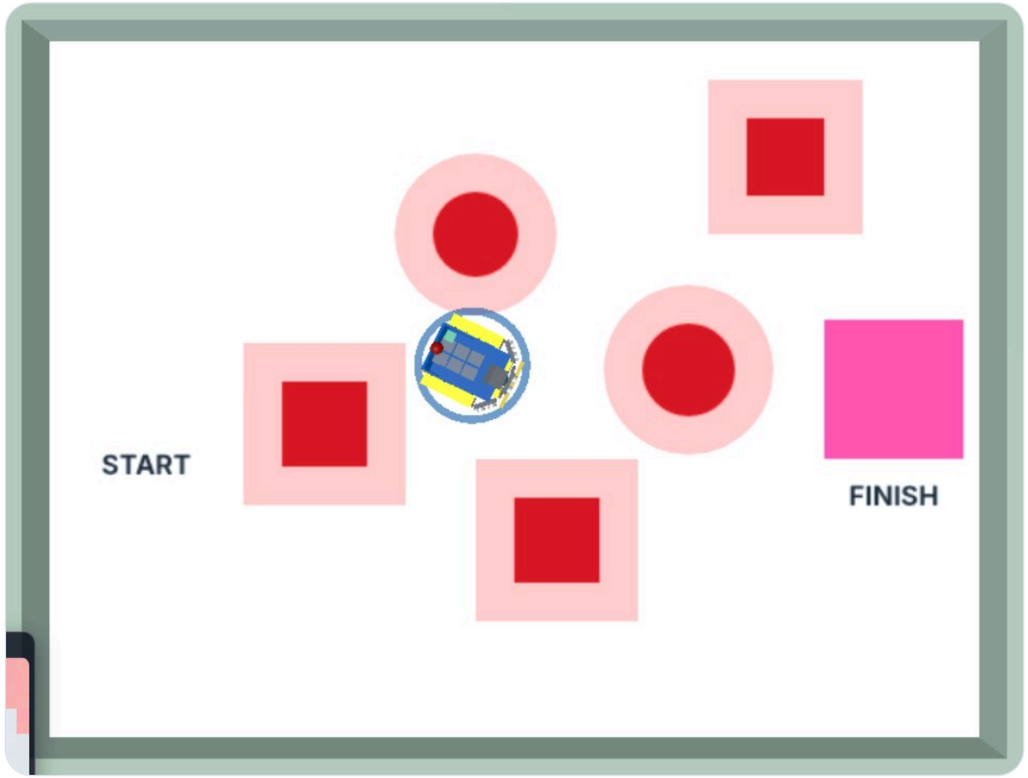
RUN · 3.5 S

First band: turn left



Red pixels: left **54**, right **77** → turn = 2.5, a left curve.

At (**72, 112**), Compass **251**.



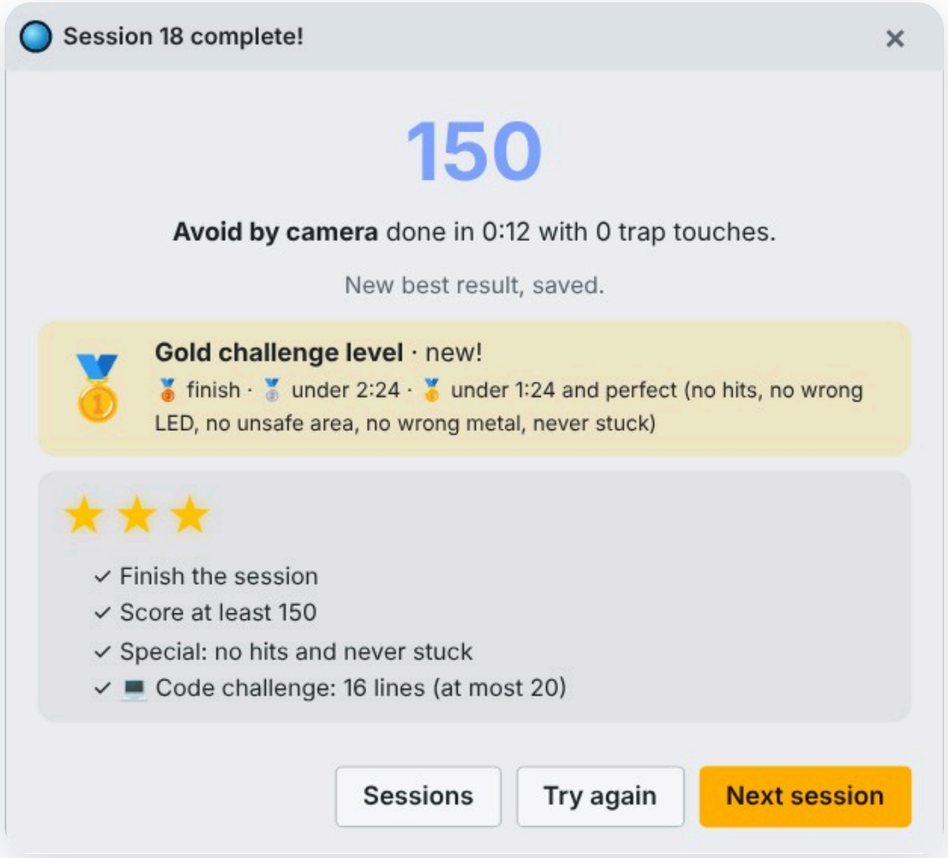
RUN · 6.8 S

Through the gap: turn right



Left **90**, right **77** → turn = −2.5, a right curve.

Then the heading pulls it on to FINISH: no touch, **0** unsafe areas.



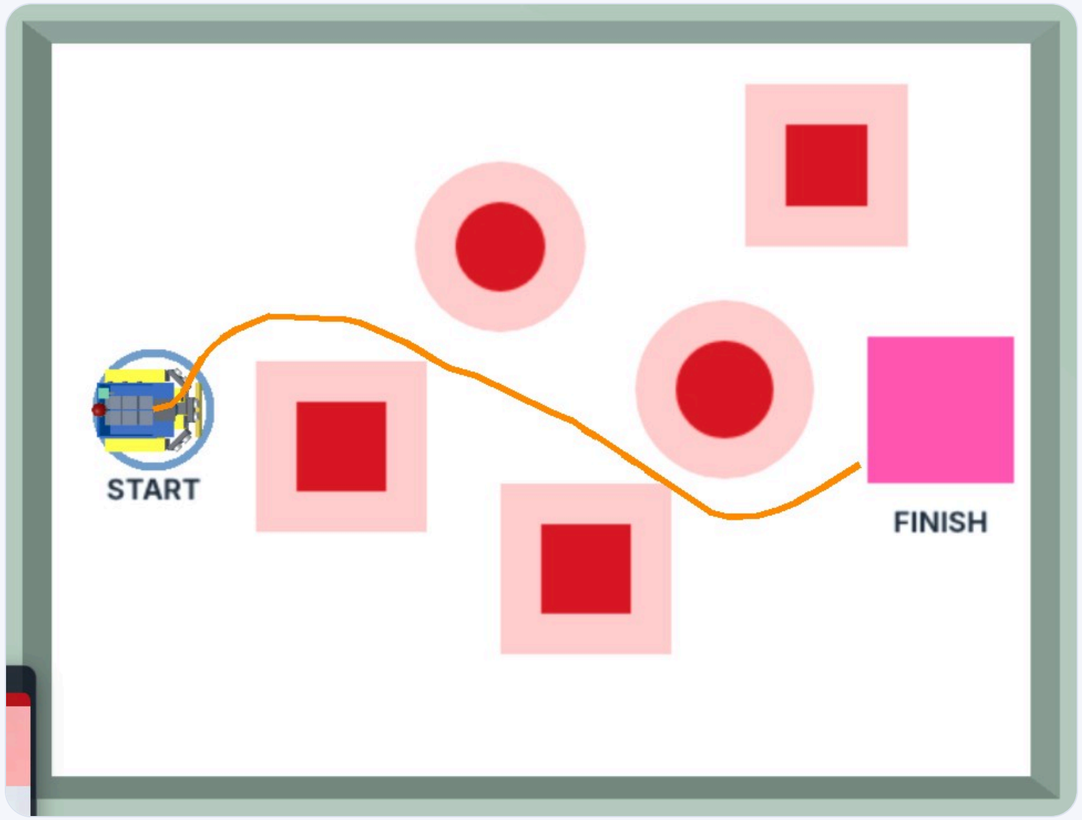
RESULT · TESTED

150 points in 0:12 · Gold

Medals · Bronze: finish · Silver: under 2:24 · Gold: under 1:24 and perfect.

Stars · finish · score 150 · no hits and never stuck.

Code challenge · at most 20 lines. We used 16.



THE PATH · RECORDED

A smooth S through the dangers

Over the first square, down through the gap, round the circle.

No zig-zag: two simple rules blend into one smooth line.

Global goal by **position**, local safety by **camera**.

DID NOT WORK?

Check first, then tune carefully

What you see	What to do
Drives into an unsafe area	Check the band test: **R > 230**, G and B **145–205**
Turns towards the red, not away	Swap the signs: left redder → **turn = −2.5**
Wanders away from FINISH	Check **atan2d(90 − PositionY, 218 − PositionX) − 90**
Wobbles at the edge of a band	Raise the threshold from 3 red pixels, or lower 2.5
Code challenge failed	Over **20 lines**? Merge short statements with ;

NEXT TIME

University Presentation 8 · Deposit by position

Collect 3 objects with the camera, then use **PositionX, PositionY and Compass** to drive home and deposit.