UNIVERSITY · PRESENTATION 5 · LEARNING WITH PYTHON

Line up and collect

Turn until the object is dead ahead, then drive straight to it: a cleaner way to use the camera.

TheBotCup 2026 · AiBotCup Simulator · University



RECAP

What the camera gave us so far

University 3

Colour tests

A pixel is gold if R, G and B are inside safe ranges.

University 4

Shape tests

O or R: the same colour, told apart by size and an empty centre.

University 5 · today

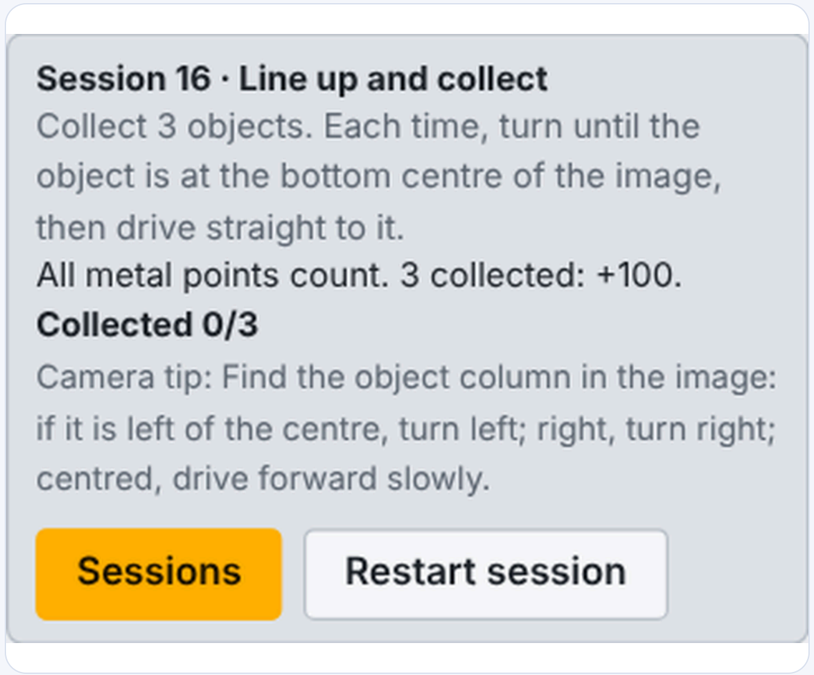
Line up first

Turn on the spot until the object is centred, then drive.

PART 1

The mission: Session 16

Collect any 3 objects



SESSION 16 · LINE UP AND COLLECT

Any 3 metals, lined up one by one

Collect any **3** objects: +**100**, plus the metal points.

Iron **+10** · gold **+20** (measured in our runs). Time limit **5:00**.

Medals · Silver: under 3:00 · Gold: under 1:45 and perfect.

Stars · finish · score 160 · no wrong metal and no wrong LED colour.



THE FIELD

Ten metals and a block

4 iron, **3 silver**, **3 gold**.

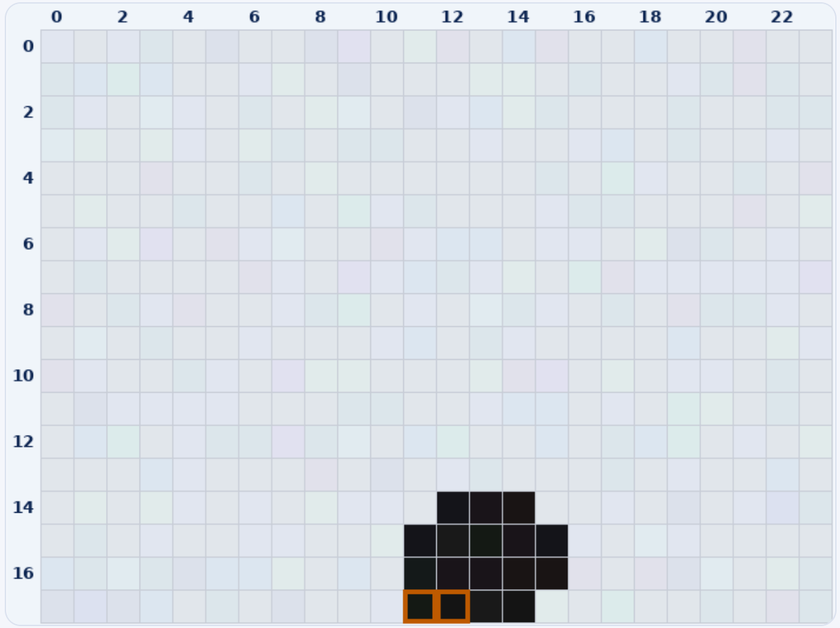
A grey **block** in the middle.

Remember the block: it will matter for the camera.

PART 2

Three ideas

Line up · choose · beware of look-alikes



IDEA 1 · LINE UP

The centre column is “straight ahead”

The image is centred on the robot’s nose: column **11.5** is straight ahead.

err = column − 11.5. Outside **±4**: turn on the spot (±1.5) until the object is in columns 8–15.

Inside ±4: drive straight, with a small correction (0.25 × err).

Line up first, drive second: the object ends up **on the pickup point**.

IDEA 2 · CHOOSE A TARGET

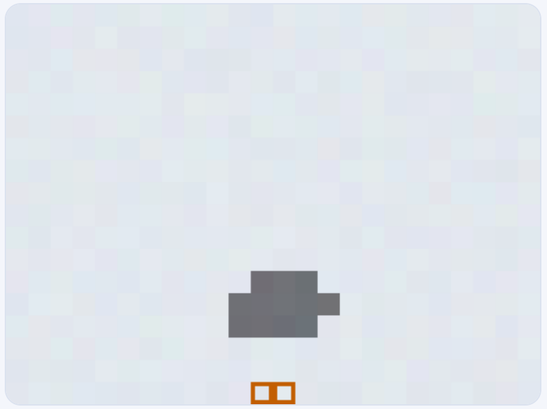
Gold first, then iron

```
col = nearest(True)                # 1st pass: gold only
if col < 0:
    col = nearest(False)           # 2nd pass: gold or iron
```

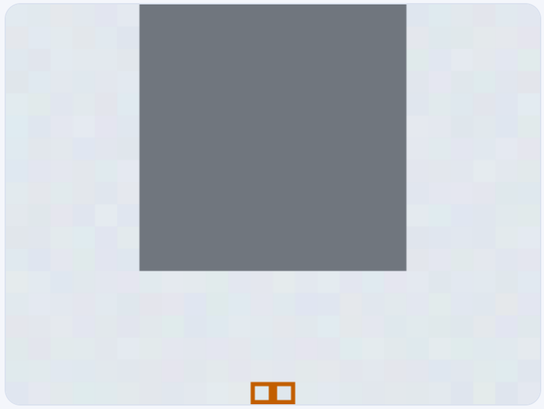
Gold is worth **20**, iron **10**: look for gold first.

If the first pass finds nothing (−1), the second accepts gold or iron.

One function, one parameter: **gold_only** switches the rule.



A **silver** object: about **110, 112, 118**



The **block**: exactly **112, 118, 126**

IDEA 3 · LOOK-ALIKES

Silver and the block: the same grey

Silver measures R 107–113, G 109–115, B 115–121. The block’s top is 112, 118, 126.

Any range wide enough for silver also catches the block.

In testing, a silver-and-iron version drove at the block and got stuck.

Our choice today: **leave silver out**. Gold and iron are enough.

PART 3

Step by step in Python

Two colour tests, a two-pass finder, line-up steering

STEP 1 · THE PLAN

Six rules, most important first

1. Both pickup pixels gold or iron → **Collect**.
2. Wall or block close (US_Front < 18) → **turn** for a random 0.3–0.8 s.
3. Nothing in view → **drive straight**.
4. Object more than 4 columns left → **turn left** on the spot.
5. More than 4 columns right → **turn right** on the spot.
6. Centred → **drive**, slower in the last 4 rows.

STEP 2 · LINES 5–9

Two colour tests

```
5  def is_gold(p):
6      return p[0] > 195 and 135 < p[1] < 200 and p[2] < 80
7
8  def is_iron(p):                              # (silver = same grey as the obstacle)
9      return p[0] < 60 and p[1] < 60 and p[2] < 60
```

Gold: the ranges from University 3.

Iron: all three below 60. Silver is left out on purpose.

STEP 3 · LINES 11–18

The two-pass finder

```
11  def nearest(gold_only):                    # column of the nearest object, or -1
12      for y in range(CAM_H - 1, -1, -1):
13          for x in range(CAM_W):
14              p = Camera[y][x]
15              if is_gold(p) or (is_iron(p) and not gold_only):
16                  gy = y
17                  return x
18      return -1
```

Bottom row first: the first hit is the closest object.

gold_only true: iron is skipped.

STEP 4 · LINES 20–33

Choose, measure, collect

```
20  def game1():
21      LED_1 = 0
22      mid = CAM_W // 2
23      col = nearest(True)                    # gold first: it is worth more
24      if col < 0:
25          col = nearest(False)
26      err = col - (CAM_W - 1) / 2            # < 0: object left of centre
27      a = Camera[CAM_H - 1][mid - 1]; b = Camera[CAM_H - 1][mid]
28      pick_a = is_gold(a) or is_iron(a)
29      pick_b = is_gold(b) or is_iron(b)
30      if pick_a and pick_b:                                  # Collect
31          WheelLeft = 0; WheelRight = 0
32          LED_1 = 2; LED_R = 255; LED_G = 0; LED_B = 0
33          Duration = 3200
```

Gold pass, then gold-or-iron pass.

err: columns away from 11.5.

Both pickup pixels metal → stop, red flashing LED, 3.2 s.

STEP 5 · LINES 34–45

Turn away, search, line up, drive

```
34      elif US_Front < 18:                                    # Wall
35          WheelLeft = 3; WheelRight = -3
36          Duration = 300 + rand(500)                         # random new direction
37      elif col < 0:                                          # Search: straight on
38          WheelLeft = 3.5; WheelRight = 3.5
39      elif err < -4:                                         # LineUpLeft
40          WheelLeft = -1.5; WheelRight = 1.5
41      elif err > 4:                                          # LineUpRight
42          WheelLeft = 1.5; WheelRight = -1.5
43      else:                                                  # Centred: drive, fine-tune
44          speed = 1.5 if gy > CAM_H - 5 else 3.5
45          WheelLeft = speed + err * 0.25; WheelRight = speed - err * 0.25
```

rand(500): a different turn each time, so the robot does not repeat one loop.

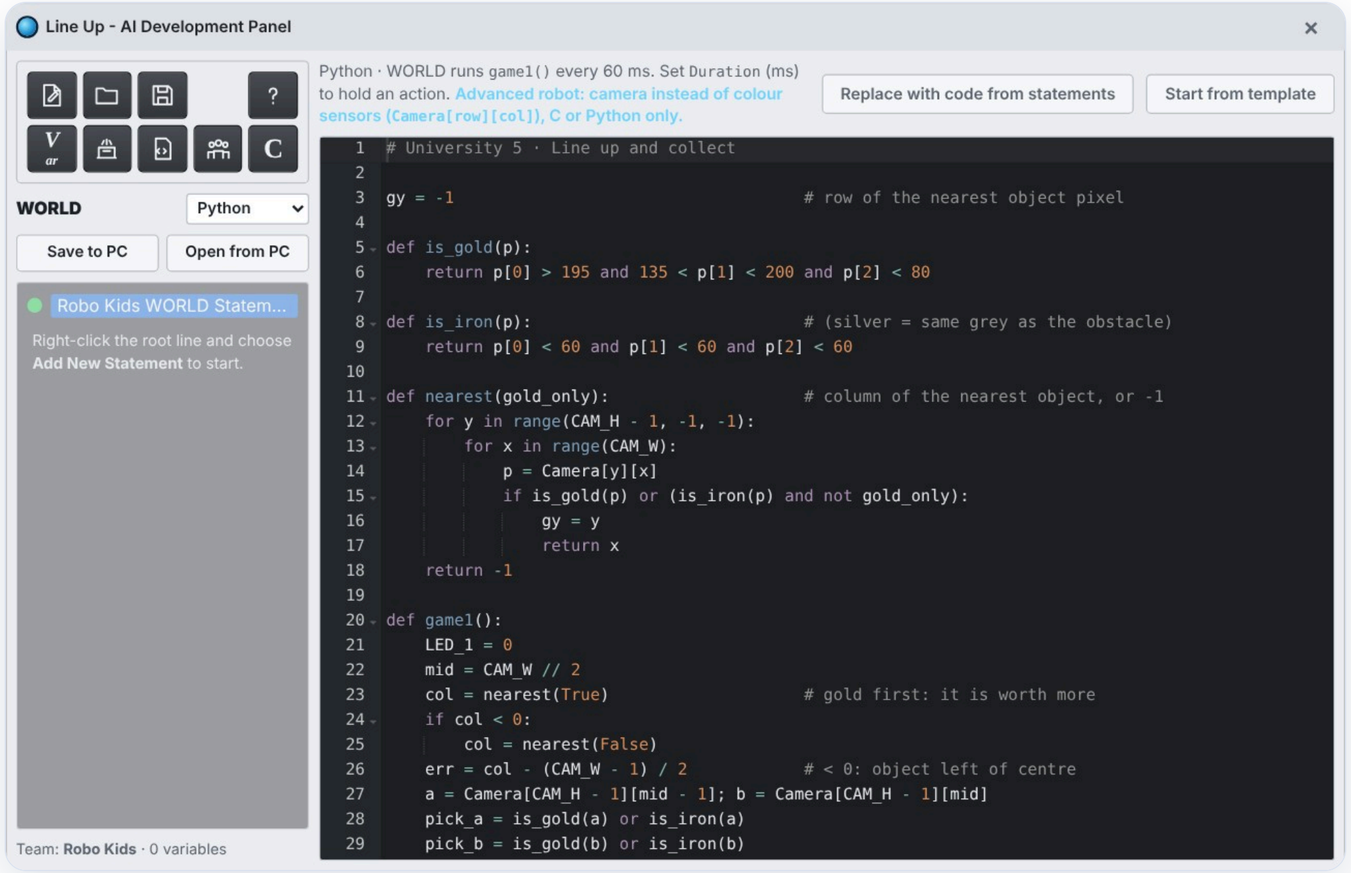
±4: the line-up zone. Turn at 1.5, slowly.

Centred: 3.5, or 1.5 in the last 4 rows.

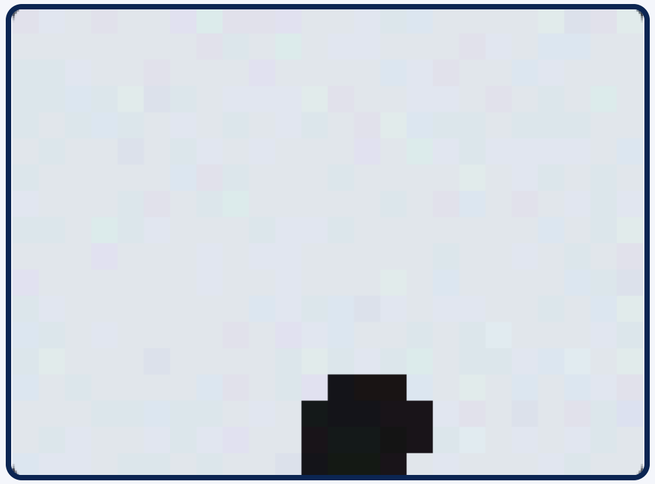
STEP 6 · BUILD AND LOAD

The whole program, built

1. Category **Advanced**, Learning, Session 16.
2. WORLD menu: **Python**.
3. Name it **Line Up**.
4. **Build**, then **Load** on BLUE.







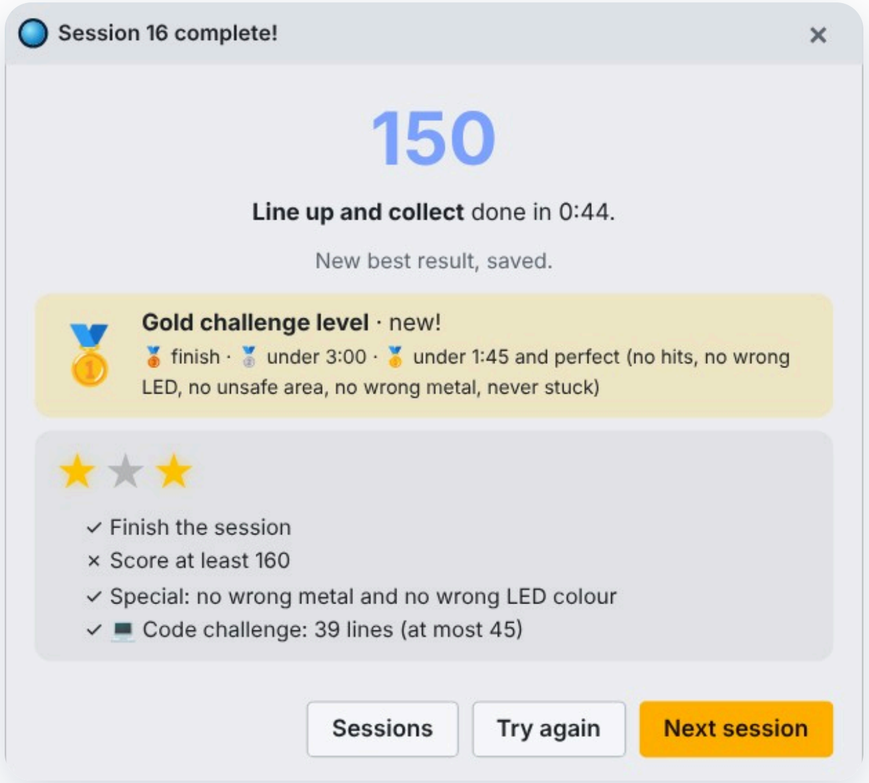
STEP 7 · RUN · 20.1 S

Lined up, on the pickup point

At **20.1 s** the first object is on the pickup point.

The robot stops and flashes red for 3.2 s.

Then: find the next one, line up, drive.



RESULT · TESTED

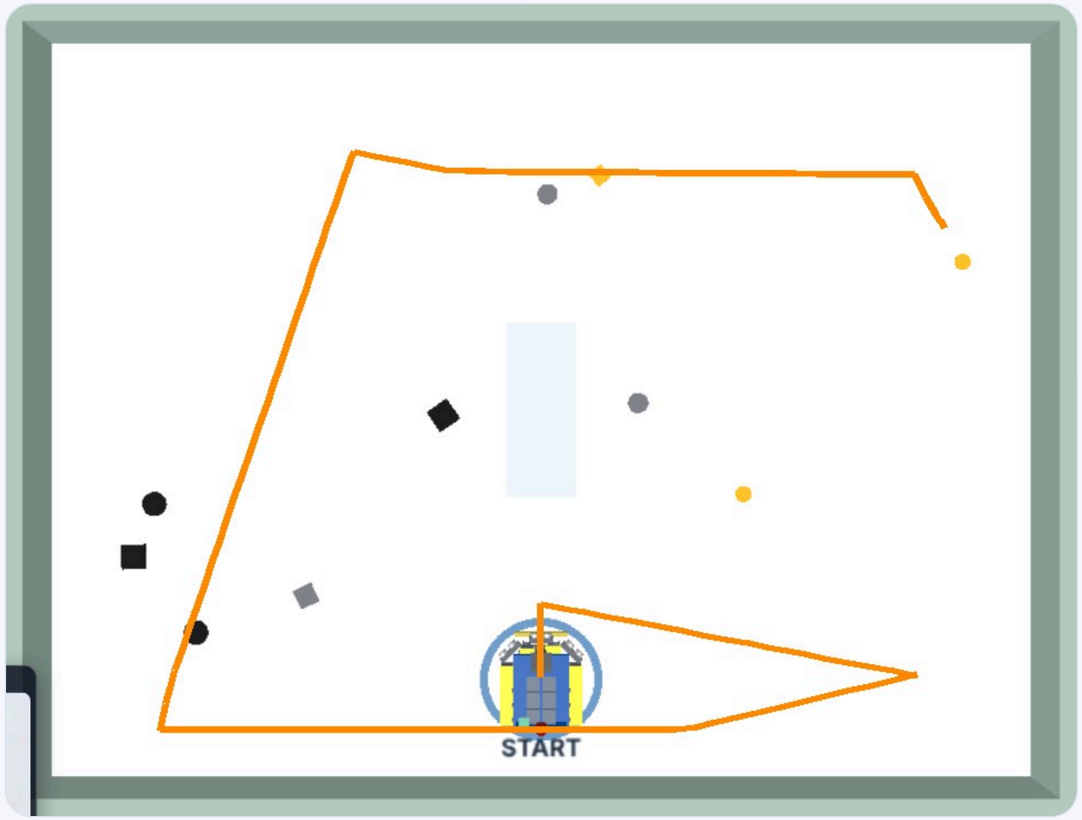
150 points in 0:44 · Gold

Medals · Silver: under 3:00 · Gold: under 1:45 and perfect.

Stars · finish · score 160 · no wrong metal and no wrong LED colour.

Code challenge · at most 45 lines. We used 39.

The score star needs **3 gold** (60 + 100). Your turn: can you get it?



THE PATH · RECORDED

Straight runs and turns on the spot

Straight runs while searching or driving to a lined-up object.

Sharp corners: the robot **lines up on the spot** before it drives.

Separate **aiming** from **driving**: simpler, more predictable.

DID NOT WORK?

Check first, then tune carefully

What you see	What to do
Turns left, right, left, right on the spot	Line-up zone too narrow: keep ±4, or turn slower
Does not turn at all when lining up	Wheel speed too low to move: at least 1.5
Drives at the block and stays there	A colour test accepts the block’s grey: no silver test
Drives the same circle for ever	Make the search straight and the wall turn random
Build error with a line number	Check colons, indentation and ( ) in the conditions

NEXT TIME

University 6 · Pond by camera

Mine inside the water pond: recognise the **pond blue** in the image and switch the LED to blue.