ANSWERS

6.1 Lines

Seeing more · Robot club · about 15 min

What this lesson is about

The line detector: cx and angle, and the empty list.

- The line detector
- Move and look again
- When there is no line
- Where the numbers come from

Questions

6 marks in all

1. With `set_cv("line")` on, what does `line()` give when a line is in view? [1 mark]

- ☐ **A** `[cx, angle]`
- ☐ **B** `[cx, cy, distance]`
- ☐ **C** `[id, cx, cy, distance]`
- ☐ **D** `[cx, cy, area, x0, y0, x1, y1, aspect]`

> **Answer:** A. Two numbers: where the line crosses the picture, and which way it is heading.

2. The robot turns right by 20 degrees while looking at a line that runs straight ahead. What happens to the line's `angle`? [1 mark]

- ☐ **A** It goes negative: the line seems to swing left
- ☐ **B** It goes positive: the line seems to swing right
- ☐ **C** It stays 0
- ☐ **D** It becomes `[]`

> **Answer:** A. It is you that turned. Turn clockwise and the line appears to head off to your left.

3. Ten centimetres ahead, one pixel is about 0.108 cm. A line has `cx = 197`. About how many cm to the right of centre is it, to one decimal place? [1 mark]

> **Answer:** 4.0 (accept within 0.2). 197 minus 160 is 37 pixels, and 37 times 0.108 is about 4 cm.

4. The detector sees about 35 cm across the strip 10 cm ahead. About how many cm to the side can the line be before it drops out of view? [1 mark]

> **Answer:** 17.5 (accept within 0.5). Half of 35 is about 17 either side of centre. Further than that and `line()` is `[]`.

5. What does this program print? [1 mark]

```
for seen in [[197, 5], []]:
    if seen:
        cx, angle = seen
        print(f"line at cx {cx}, angle {angle}")
    else:
        print("no line")
```

> **Answer:**
>
> ```
> line at cx 197, angle 5
> no line
> ```
>
> An empty list counts as false, so the second reading prints the fallback. Every line program has to cope with `[]`.

6. The task starter just calls `print(line())`. What is it missing? [1 mark]

- ☐ **A** `set_cv("line")`, to switch on the line detector
- ☐ **B** `set_cv("apriltag")`
- ☐ **C** `forward(50)`, because the line detector only works when moving
- ☐ **D** Nothing, `line()` always works

> **Answer:** A. The camera runs one detector at a time, and you have to pick it first, just as with tags in Module 4.

The task: see the line

Print `line at cx <cx>, angle <angle>` from the detector. Do not drive.

```
# the two lines every program starts with: the commands, then the robot
from bugbot import *
connect()

print(line())
```

The hint students can ask for: Switch the camera to the line detector and print where the line is, like `line at cx 197, angle 0.0`. Do not drive.

A solution

```
from bugbot import *
connect()
set_cv('line')
cx, angle = line()
print(f'line at cx {cx}, angle {angle}')
```

Any program that meets the task's checks is marked correct in the simulator; this is one way, not the only way.