WORKSHEET

2.5 Drift and correction

BugBotLab

Sensing · Robot club · about 20 min

Name _____ Class _____ Date _____

What this lesson is about

Why the robot wanders, and the first closed loop.

- Watch the drift
- Open loop and closed loop
- The simplest correction
- Correcting the heading too
- Steering while moving

Questions

7 marks in all

1. Which of these programs is closed loop? [1 mark]

- ☐ **A** One that checks position() after each leg and slides back to the line
- ☐ **B** forward(70), wait(3), stop()
- ☐ **C** forward(70, distance=64) on its own
- ☐ **D** A for loop that drives the same square every time

2. Why does the robot slide sideways when it drives forward? Choose all that apply. [1 mark]

Tick every answer that is true.

- ☐ **A** The four feet do not push exactly equally
- ☐ **B** The mat is not perfectly even
- ☐ **C** Tiny errors add up over a long drive
- ☐ **D** There is a bug in forward()
- ☐ **E** position() resets itself as it drives

3. After a 10 cm leg, `position()` says x is 2.5. What does the correcting program do? [1 mark]

```
x, y = position()
if x > 1:
    left(40, distance=x)
elif x < -1:
    right(40, distance=-x)
```

- ☐ **A** Slides left 2.5 cm
- ☐ **B** Slides right 2.5 cm
- ☐ **C** Nothing, because 2.5 is too small
- ☐ **D** Turns left 2.5 degrees

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

```
error = 355
if error > 180:
    error = error - 360
print(error)
```

5. In `drive(70, -x * 15, 0)`, `x` is 2. What is the sideways value? Give a whole number. [1 mark]

6. You change `-x * 15` to `-x * 40`. What do you expect? [1 mark]

- ☐ **A** It settles closer to the line
- ☐ **B** It drifts further off the line
- ☐ **C** It drives faster
- ☐ **D** Nothing changes

7. Why is it worth correcting the heading as well as x? [1 mark]

- ☐ **A** A wrong heading turns into a sideways error on the next leg
- ☐ **B** heading() is the only thing that drifts
- ☐ **C** Correcting x resets the heading
- ☐ **D** It is not, x is all that matters

The task: straight and true

Drive about 64 cm and finish inside the small square, still facing 0 within 6 degrees. Correct as you go.

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

# drive forward at 70 for 64 cm, then stop
forward(70, distance=64)
```

Plan your program here, then type it in and press Run.

Do it on the robot

www.bugbotlab.com/learn/2-5-drift-and-correction/

The simulator checks it and tells you when it passes. Nothing to install, no account.

Challenges

1. Make the straight-line task pass at speed 100.
2. Combine heading correction and sideways correction in one loop and time it against the two-step version.
3. Print the largest `x` error you saw during the drive.