ANSWERS

U10.3 Cross-track error

Following a trajectory · University · about 35 min

What this lesson is about

Splitting the error into along and across the path, and a controller for the half that matters.

- The path frame
- Controlling it
- Choosing the gain
- The dead band, again

Questions

7 marks in all

1. A leg runs from (0, 0) to (30, 40) and the robot is at (10, 20). What does this print? [1 mark]

```
import math
ax, ay, bx, by = 0.0, 0.0, 30.0, 40.0
x, y = 10.0, 20.0
length = math.hypot(bx - ax, by - ay)
ux, uy = (bx - ax) / length, (by - ay) / length
nx, ny = -uy, ux
along = (x - ax) * ux + (y - ay) * uy
cross = (x - ax) * nx + (y - ay) * ny
print(round(along, 2), round(cross, 2))
```

Answer:

```
22.0 4.0
```

u = (0.6, 0.8) and the left normal is (-0.8, 0.6). along = 6 + 16 = 22 cm, cross = -8 + 12 = +4 cm, so the robot is 4 cm to the left of the path.

2. The line runs from (60, 30) to (60, 170) and the robot, facing +y, is at (70, 50). With correction = clamp(-0.9 x cross, 12) along the left normal, and drive()'s sideways term positive to the robot's right at 15 cm/s for 100, what sideways term should be sent? [1 mark]

- ☐ **A** -60, because the correction of +9 cm/s is along the left normal (-1, 0), which is the robot's left
- ☐ **B** +60, the correction of +9 cm/s converted to percent and sent straight in
- ☐ **C** +9, the correction in cm/s
- ☐ **D** -80, the clamped maximum

Answer: A. cross = (70 - 60) x (-1) = -10, so correction = +9 cm/s along (-1, 0). The sideways term is 100 x 9 x nx / 15 = -60. Sending +60 straight in pushes the robot further from the line.

3. The holonomic cross-track loop has gain k = 0.9 per second and the robot moves along the path at 11 cm/s. Over how many centimetres of travel is one time constant of the correction spread? Give 1 decimal place. [1 mark]

Answer: 12.2 (accept within 0.2). The time constant is 1 / k = 1.11 s, and 1.11 s at 11 cm/s is 12.2 cm of travel.

4. The robot sits 1 cm off the path with k = 0.9. What percentage command does the correction ask for, given 100 percent is 15 cm/s sideways? [1 mark]

> **Answer:** 6 (accept within 0.1). 0.9 x 1 = 0.9 cm/s, and 100 x 0.9 / 15 = 6 percent, well inside the 15 percent dead band, so nothing moves.

5. The holonomic cross-track loop is first order and cannot oscillate on its own. Why can it still wobble in practice? [1 mark]

- ☐ **A** Delays: the 0.25 s drive lag, the loop period and the dead band
- ☐ **B** Adding a sideways velocity makes the loop second order
- ☐ **C** The cross-track error is signed
- ☐ **D** The along-track error feeds into the cross-track error

> **Answer:** A. Velocity to position is one integrator. Everything that makes it oscillate is a delay, which adds phase lag.

6. What happens as the cross-track gain k is raised well above 0.9? [1 mark]

Tick every answer that is true.

- ☐ **A** The correction saturates at the available sideways speed, so the response stops getting faster
- ☐ **B** The phase lost to the 0.25 s lag starts to matter and the approach becomes a wobble
- ☐ **C** The loop becomes second order, like a car's steering
- ☐ **D** The time constant 1 / k gets longer

> **Answer:** A, B. Only 15 cm/s of sideways speed exists, and the lag costs phase. The loop order does not change, and 1 / k gets shorter, not longer.

7. Why does the Stanley controller for a car divide the cross-track term by the speed? [1 mark]

- ☐ **A** At higher speed the geometry closes a given error sooner, so the same error needs a gentler steering correction
- ☐ **B** Faster cars have larger cross-track errors, so the term needs shrinking
- ☐ **C** It converts the error from centimetres to degrees
- ☐ **D** It stops the steering saturating at low speed

> **Answer:** A. Without the speed scaling, a gain that is calm at walking pace becomes unstable on the motorway.

The task: drive the cross-track error to zero

The taped line runs from (60, 30) to (60, 170). The robot starts at (85, 30), which is 25 cm to the right of it. Get onto the line, drive along it to the far end, plot `cross track` the whole way, and stop near (60, 163).

```
from bugbot import *
import math
connect()

DT = 0.1
START = (85.0, 30.0)
AX, AY, BX, BY = 60.0, 30.0, 60.0, 170.0
```

The hint students can ask for: The taped line runs from (60, 30) to (60, 170) and the robot starts 25 cm to the right of it. Build a unit vector along the path and one at right angles to it, split the error into along and across, and add a sideways velocity proportional to the across part while you drive along.

A solution

```
from bugbot import *
import math
connect()

DT = 0.1
V_MAX, V_LAT = 20.0, 15.0
START = (85.0, 30.0)
AX, AY, BX, BY = 60.0, 30.0, 60.0, 170.0
CRUISE, K_CROSS = 11.0, 0.9

ux, uy = BX - AX, BY - AY
LENGTH = math.hypot(ux, uy)
ux, uy = ux / LENGTH, uy / LENGTH            # along the path
nx, ny = -uy, ux                             # across it

def band(u):
    """below about 15 percent the drive does nothing at all, so ask for a little more or
for nothing"""
    if abs(u) < 5.0:
        return 0.0
    return max(17.0, min(100.0, u)) if u > 0 else min(-17.0, max(-100.0, u))

def steer(wx, wy):
    """a velocity in the world, through the inverse kinematics, as a command"""
    h = math.radians(heading())
    vx = wx * math.cos(h) - wy * math.sin(h)
    vy = wx * math.sin(h) + wy * math.cos(h)
    e = (0.0 - heading() + 180) % 360 - 180
    rot = 0.0 if abs(e) < 3 else max(-30.0, min(30.0, 2.0 * e))
    drive(band(100 * vy / V_MAX), band(100 * vx / V_LAT), band(rot))

cross = 0.0
for tick in range(500):
    px, py = position()
    x, y = START[0] + px, START[1] + py
    along = (x - AX) * ux + (y - AY) * uy
    cross = (x - AX) * nx + (y - AY) * ny
    plot("cross track", cross)
```

```
    if along > LENGTH - 5:
        break
    speed = min(CRUISE, 0.8 * (LENGTH - along))
    correct = max(-12.0, min(12.0, -K_CROSS * cross))
    steer(ux * speed + nx * correct, uy * speed + ny * correct)
    wait(DT)
stop()
wait(0.4)
print("cross track:", round(cross, 2))
```

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