ANSWERS

U10.5 Feedforward and feedback

Following a trajectory · University · about 30 min

What this lesson is about

A ramp leaves a proportional controller permanently behind, and the model knows how to fix it.

- The arithmetic
- The better answer
- Where the model comes from
- What feedforward cannot do

Questions

6 marks in all

1. A proportional controller with K = 1.5 per second chases a reference moving at 10 cm/s. How far behind does it settle, in cm to 2 decimal places? [1 mark]

> **Answer:** 6.67 (accept within 0.01). In steady state K e = v, so e = 10 / 1.5 = 6.67 cm, for as long as the ramp lasts.

2. The reference speed is now fed forward, but the model is 10 percent out. With the same v = 10 cm/s and K = 1.5, what steady error is left, in cm to 2 decimal places? [1 mark]

> **Answer:** 0.67 (accept within 0.01). Feedback only has the model error to correct: 0.1 x 10 / 1.5 = 0.67 cm, ten times smaller.

3. Why can a proportional controller never catch a reference moving at constant speed? [1 mark]

☐ **A** It only produces a command when there is an error, and the command must stay non-zero to keep moving

☐ **B** Its gain is always too low

☐ **C** The drive's dead band stops it moving at constant speed

☐ **D** The reference is sampled too slowly

> **Answer:** A. A non-zero command for ever needs a non-zero error for ever. Raising K shrinks it but never removes it.

4. This simulates a robot whose real speed is 90 percent of the model's, chasing a 10 cm/s ramp, first with gain alone and then with the reference fed forward. What does it print? [1 mark]

```
V, K, DT = 10.0, 1.5, 0.1
for ff in (0.0, 1.0):
    pos = 0.0
    for tick in range(300):
        ref = V * tick * DT
        err = ref - pos
        pos += 0.9 * (ff * V + K * err) * DT
    print(ff, round(err, 2))
```

Answer:

```
0.0 7.41
1.0 0.74
```

In steady state 0.9 (ff V + K e) = V. Gain alone gives e = 11.11 / 1.5 = 7.41 cm; with feedforward only the 1.11 cm/s shortfall is left, e = 0.74 cm.

5. Which statements about feedforward and feedback are right? [1 mark]

Tick every answer that is true.

- ☐ **A** Feedforward has no stability cost because it is not in the loop
- ☐ **B** Feedforward alone drifts, because nothing measures the result
- ☐ **C** Feedback handles what the model does not know, such as a slope or a flat battery
- ☐ **D** With good feedforward the feedback loop can be removed
- ☐ **E** Feedforward reduces the steady error by raising the loop gain

Answer: A, B, C. Feedforward makes it happen and feedback fixes what it got wrong. Feedforward does not change the loop gain, and it cannot see what actually happened.

6. A robot running feedforward plus feedback is stuck against a chair leg. What does the feedforward term do? [1 mark]

- ☐ **A** It keeps commanding cruise speed, because it never sees what actually happened
- ☐ **B** It drops to zero, because the error has grown
- ☐ **C** It reverses, because the model predicts a collision
- ☐ **D** It doubles, to push past the obstacle

Answer: A. Feedforward depends only on the reference. Anything that depends on the result has to come from the feedback path.

The task: how far behind a ramp

Drive the same 10 cm/s ramp twice: once with proportional control alone, once with the reference speed fed forward as well. Average the size of the error over the last few seconds of each leg, and print `p only:` and `with ff:`. Plot `ref`, `actual` and `error`.

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

DT = 0.1
V_REF, K = 10.0, 1.5
```

The hint students can ask for: Run the same ramp twice: once commanding only the gain times the error, once adding the reference's own speed through the kinematics before the correction. Average the size of the error over the last few seconds of each leg, when the transient has died away.

A solution

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

DT = 0.1
V_MAX = 20.0
V_REF, K = 10.0, 1.5

def band(u):
    return 0.0 if abs(u) < 3.0 else max(-100.0, min(100.0, u))

def leg(seconds, feedforward, name):
    """drive a ramp reference for this long, and report the settled following error"""
    base = position()[1]
    errors = []
    for tick in range(int(seconds / DT)):
        t = tick * DT
        ref = V_REF * t
        here = position()[1] - base
        err = ref - here
        want = (V_REF if feedforward else 0.0) + K * err
        drive(band(100 * want / V_MAX), 0, 0)
        plot("ref", base + ref)
        plot("actual", position()[1])
        plot("error", err)
        if t > seconds - (4.0 if feedforward else 5.0):
            errors.append(abs(err))
        wait(DT)
    stop()
    wait(0.6)
    return sum(errors) / len(errors)

# the gain alone: the reference runs away and the error settles where the gain can just
keep up
print("p only:", round(leg(9.0, False, "p"), 2))
# the same ramp with the reference speed handed straight to the motors
print("with ff:", round(leg(7.0, True, "ff"), 2))
```

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