ANSWERS

U6.2 Predict and correct

State estimation · University · about 30 min

What this lesson is about

The shape every filter has: a model that guesses forward, a measurement that pulls it back.

- On the robot
- Why predict at all
- What a model is
- The residual

Questions

7 marks in all

1. Put one tick of the lesson's fusion loop in order. [1 mark]

Number the lines 1 to 4 to put them in the right order.

___ `measured = START_GAP - distance()`
___ `est = 0.9 * est + 0.1 * measured`
___ `plot("estimate", est)`
___ `est += flow()[1] * DT`

Answer:

```
est += flow()[1] * DT
measured = START_GAP - distance()
est = 0.9 * est + 0.1 * measured
plot("estimate", est)
```

Predict with the motion measurement, take the measurement, correct towards it, then plot.

2. Three ticks of predict and correct with made-up sensor values. What does it print? [1 mark]

```
DT, START_GAP = 0.1, 110.0
est = 0.0
for speed, d in ((20.0, 107.0), (20.0, 103.0), (20.0, 104.0)):
    est += speed * DT
    measured = START_GAP - d
    est = 0.9 * est + 0.1 * measured
    print(round(est, 2))
```

Answer:

```
2.1
4.39
6.35
```

Tick 1 predicts 2, measures 3, and blends to 2.1. Tick 2 predicts 4.1, measures 7, blends to 4.39. Tick 3 predicts 6.39, measures 6, blends to 6.35.

3. What is measured - est called? [1 mark]

Answer: innovation. The innovation, or residual, is the part of the measurement the model did not predict, and it is the most useful diagnostic a filter has.

4. The plotted innovation is consistently positive. What does that say? [1 mark]

- ☐ **A** The model is biased and the filter is fighting it
- ☐ **B** The filter is working well
- ☐ **C** The measurement noise is larger than expected
- ☐ **D** The blend weight is too large

> **Answer:** A. Innovations should scatter around zero. A consistent sign means the prediction is systematically off, for example a flow scale error.

5. Why does predict and correct keep up with a moving robot better than a low pass filter on the distance alone? [1 mark]

- ☐ **A** The prediction uses a measurement of the motion, so the correction only fixes a small residue
- ☐ **B** It uses a smaller alpha
- ☐ **C** It uses two sensors, which halves the noise
- ☐ **D** It runs the loop faster

> **Answer:** A. A low pass filter smooths towards an average and lags a moving target. The model knows the robot is moving, so it does most of the work.

6. The correct step is changed to est = 0.98 * est + 0.02 * measured. What is the filter now trusting? [1 mark]

- ☐ **A** Mostly the flow prediction, so drift is corrected only slowly
- ☐ **B** Mostly the wall measurement, so the estimate is noisy
- ☐ **C** Both equally
- ☐ **D** Neither, the estimate stops updating

> **Answer:** A. The measurement gets 2 percent per tick, so the estimate is smooth and follows the flow sensor, including its drift, for longer.

7. The robot has no velocity sensor. What does the lesson suggest as the model for the predict step? [1 mark]

- ☐ **A** The commands sent, through the kinematics from U2
- ☐ **B** A copy of the last measurement
- ☐ **C** A low pass filter on the measurement
- ☐ **D** No prediction; correct only

> **Answer:** A. The prediction is where knowledge of the machine lives. Without a velocity sensor, the commands plus the kinematics say how the state should change.

The task: predict, then correct

Drive at least 40 cm towards the wall, running predict and correct, plotting `measured` and `estimate`, and print `my y:`. `position()` is not allowed.

```
from bugbot import *
connect()

DT = 0.1
START_GAP = 110.0
est = 0.0
```

The hint students can ask for: Predict with the flow sensor: estimate += flow()[1] * dt. Correct with the wall: the robot started 110 cm from the wall's face, so the depth reading says you have travelled 110 - distance(). Blend the two, and plot both so you can see the noisy one and the steady one together.

A solution

```
from bugbot import *
connect()

DT = 0.1
START_GAP = 110.0          # cm from the robot to the wall's face at the start
est = 0.0

forward(60)
for i in range(70):
    est += flow()[1] * DT                      # predict
    measured = START_GAP - distance()          # the same quantity, measured
    est = 0.9 * est + 0.1 * measured           # correct
    plot("measured", measured)
    plot("estimate", est)
    if est > 55:
        stop()
    wait(DT)
stop()
print("my y:", round(est, 1))
```

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