WORKSHEET

U7.4 Weighing the guesses

BugBotLab

Localisation · University · about 35 min

Name ____________ Class ____________ Date ____________

What this lesson is about

The measurement update: how likely is this reading if the robot were there?

- The sensor model is the interesting part
- Normalising, and the effective sample size
- Do not lie about sigma

Questions

7 marks in all

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

```
import math
sigma = 3.0
measured = 144.0
for predicted in (144.0, 147.0, 150.0):
    d = predicted - measured
    w = math.exp(-d * d / (2 * sigma * sigma))
    print(predicted, round(w, 3))
```

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

```
weights = [0.4, 0.3, 0.2, 0.1]
neff = 1.0 / sum(w * w for w in weights)
print(round(neff, 2))
```

3. A cloud has 100 particles. After normalising, two particles have weight 0.5 each and every other particle has weight 0. What is the effective sample size? [1 mark]

4. Why add a tiny floor such as 1e-12 to every weight? [1 mark]

- ☐ **A** One unexpected reading could otherwise make every weight zero, and normalising would divide by zero
- ☐ **B** It stops the weights summing to more than one
- ☐ **C** It makes the likelihood a proper Gaussian
- ☐ **D** It raises the effective sample size to N after every update

5. The sensor's real noise has a standard deviation of 3 cm, but the filter uses sigma = 0.5 cm. What happens? [1 mark]

☐ **A** The filter becomes over-confident and can throw away the true pose because of one unlucky reading
☐ **B** The filter becomes more accurate, because a smaller sigma sharpens the estimate
☐ **C** Nothing changes, because the weights are normalised afterwards
☐ **D** The effective sample size rises, because more particles fit the reading

6. What is the quantity 1 / (sum of the squared normalised weights) called? [1 mark]

7. The robot is 80 cm from the far wall and has turned 60 degrees towards the left wall. The depth sensor reads 152 cm. A filter weighs its cloud with predicted = 200 - y. What happens? [1 mark]

☐ **A** The guesses near y = 48 win, about 70 cm from the truth, because 200 - y is only right while the robot faces the far wall square on
☐ **B** Nothing goes wrong, because the weights are normalised afterwards
☐ **C** Every weight becomes exactly zero and the filter stops
☐ **D** The cloud stays where it was, because a slanted reading carries no weight

The task: weigh the guesses

Scatter particles over the mat, take a reading, weight them, and print `best y:` (the weighted mean) and `neff:`.

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

N, SIGMA = 500, 3.0
particles = [random.uniform(0, 200) for i in range(N)]
```

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

Do it on the robot
www.bugbotlab.com/learn/u7-4-weighing-the-guesses/
The simulator checks it and tells you when it passes. Nothing to install, no account.

Challenges

1. Use sigma = 0.5 and look at neff. How many particles survive?

2. Weight the cloud against two readings taken from the same spot. Does neff fall further?
3. Plot the weights against the particle's y. What shape is it?