WORKSHEET

U3.1 Integrating velocity

Odometry and drift · University · about 25 min

BugBotLab

Name _____ Class _____ Date _____

What this lesson is about

The odometry update, one step at a time, and what the step size costs you.

- The simplest case
- Reason one: the step is not an instant
- Reason two: the reading is not the speed
- Reason three: the loop is not exactly 0.1 s

Questions

6 marks in all

1. Speeds in cm/s are read every 0.1 s as the robot speeds up. This integrates them two ways. What does it print? [1 mark]

```
speeds = [0.0, 8.0, 14.0, 18.0, 20.0, 20.0]
DT = 0.1
euler = 0.0
trap = 0.0
for i in range(1, len(speeds)):
    euler += speeds[i] * DT
    trap += 0.5 * (speeds[i] + speeds[i - 1]) * DT
print("euler:", round(euler, 2))
print("trapezium:", round(trap, 2))
```

2. Which integration method takes the speed read at one instant and uses it as if it were the average over the whole step? [1 mark]

3. When do the trapezium rule and Euler integration give the same answer? [1 mark]

- ☐ **A** Once the speed is steady
- ☐ **B** While the robot is speeding up
- ☐ **C** Only when the step is very small
- ☐ **D** Never, because the trapezium rule always averages two readings

4. A loop multiplies by DT = 0.1, but its real period is 0.11 s. The robot drives at a steady 20 cm/s for 50 passes. By how many centimetres does the integral fall short of the true distance? [1 mark]

5. Why does assuming the loop period, rather than measuring it with clock(), cause an error that grows through the run? [1 mark]

- ☐ **A** The real period is always at least the wait, so every step is short the same way and the errors add up
- ☐ **B** Assumed periods are noisy, and noise adds up as a random walk
- ☐ **C** clock() drifts less than wait(), so it is only needed on long runs
- ☐ **D** The flow sensor's scale changes if the period is not exact

6. A straight-line integral of flow() is close to the truth but not exact. Which of these are reasons the lesson gives? [1 mark]

Tick every answer that is true.

- ☐ **A** The reading is one instant, used as the average over the step
- ☐ **B** The flow reading has noise and a scale a few percent off
- ☐ **C** The loop period is not exactly the 0.1 s it is multiplied by
- ☐ **D** The deadman stops the robot between readings
- ☐ **E** position() is only accurate to a few centimetres

The task: integrate a velocity

Drive at least 50 cm in a straight line, integrating `flow()` as you go, and print your own answer as `my y: 54.3`. It is marked against where the robot really ended up.

```
from bugbot import *
connect()

DT = 0.1
y = 0.0
forward(70)
```

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

Do it on the robot

www.bugbotlab.com/learn/u3-1-integrating-velocity/

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

Challenges

1. Do the same run with the trapezium rule and compare the two estimates against the truth.

2. Measure the real loop period with `clock()` and use that instead of the constant. How much does it change?
3. Integrate with `DT = 0.5`. Where does the error come from now?
