ANSWERS

U2.3 A drive that goes sideways

BugBotLab

Kinematics and frames · University · about 25 min

What this lesson is about

Three degrees of freedom on the floor, and what holonomic buys you.

- What that buys you
- The three commands are not the same size

Questions

6 marks in all

1. Why is a car non-holonomic while the BugBot is holonomic? [1 mark]

- ☐ **A** A car has three degrees of freedom on the floor but only two controls, and cannot move sideways directly
- ☐ **B** A car has fewer degrees of freedom on the floor than the BugBot
- ☐ **C** A car cannot change its heading while it is moving
- ☐ **D** A car's controls are blocking and the BugBot's are not

> **Answer:** A. Both have x, y and heading. The BugBot has three independent controls for them, so any combination of forward, sideways and turning is available at once.

2. This robot's sideways full scale is 15 cm/s. At what speed does drive(0, 70, 0) move it, in cm/s? [1 mark]

> **Answer:** 10.5 (accept within 0.05). Command 70 is 70 percent of that axis's own full scale: 0.7 × 15 = 10.5 cm/s, not 70 percent of the forward axis's 20.

3. With V_MAX = 20 cm/s and V_LAT = 15 cm/s, drive(70, 70, 0) is sent expecting a 45 degree diagonal. At what angle to the right of straight ahead does the robot actually travel, to the nearest degree? [1 mark]

> **Answer:** 37 (accept within 0.5). Forward is 0.7 × 20 = 14 cm/s and sideways is 0.7 × 15 = 10.5 cm/s. The angle is atan(10.5 / 14) = 36.9 degrees, so equal commands do not give equal speeds.

4. Which of these does a holonomic drive buy you? [1 mark]

Tick every answer that is true.

- ☐ **A** The robot can point its camera at one thing while travelling towards another
- ☐ **B** Errors in x and y can be corrected by separate terms at the same time
- ☐ **C** Any curve on the floor is a path the robot can follow
- ☐ **D** Better efficiency than a wheeled car
- ☐ **E** Fewer actuators to control

> **Answer:** A, B, C. Position and heading come apart, and planning and control both get simpler. The costs are more actuators, worse efficiency and a drive that is harder to model.

5. Code that treats a command as a velocity asks for a straight diagonal. What happens? [1 mark]

- ☐ **A** The robot drifts to one side of the line, because each axis has its own full scale
- ☐ **B** The robot follows the line exactly, just more slowly
- ☐ **C** The robot spins, because the diagonal couples into rotation
- ☐ **D** Nothing goes wrong, because commands on every axis are percentages of the same speed

> **Answer:** A. The error is the ratio between the axes' full scales, and it is different in x and y, so the direction is wrong, not just the speed.

6. The green zone is to the robot's right. Which command gets it there without turning? [1 mark]

- ☐ **A** drive(0, 70, 0)
- ☐ **B** drive(70, 0, 0)
- ☐ **C** drive(0, 0, 70)
- ☐ **D** drive(0, -70, 0)

> **Answer:** A. drive takes forward, sideways, rotation, and positive sideways is to the robot's right. The heading stays where it started.

The task: crab into the zone

The green zone is to the robot's right. Get into it without turning: the robot has to arrive facing the same way it started.

```
from bugbot import *
connect()

# drive(forward, sideways, rotation)
```

The hint students can ask for: The zone is to the robot's right, and it must arrive facing the same way it started. drive(fwd, lat, rot) takes a sideways term, and right() is the blocking version of it.

A solution

```
from bugbot import *
connect()

while position()[0] < 62:
    drive(0, 80, 0)
    wait(0.1)
stop()
```

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