WORKSHEET

U10.4 Pure pursuit

Following a trajectory · University · about 35 min

Name _____ Class _____ Date _____

What this lesson is about

Chase a point a fixed distance ahead on the path, and choose the one number that decides everything.

- Why it works
- The trade-off
- Speed-dependent look-ahead
- Finding the look-ahead point

Questions

8 marks in all

1. A car chases a look-ahead point 18 cm ahead that is 3 cm to the side in the body frame. What radius of arc does pure pursuit command, in cm? [1 mark]

2. What does a longer look-ahead distance cost? [1 mark]

- ☐ **A** Corners are cut, and on a tight bend the robot can take the chord straight across the inside
- ☐ **B** The robot weaves down straight lines
- ☐ **C** The controller needs a higher gain to stay on the path
- ☐ **D** The look-ahead point can move backwards along the path

3. The BugBot follows a circle of radius 40 cm with pure pursuit, pointing its velocity at a look-ahead point L = 18 cm further along. By roughly how much does it track inside the circle, in cm to 1 decimal place? [1 mark]

4. A robot that steers like a car, turning to face the look-ahead point and driving forwards, weaves down a straight line with a period of about 2 s. What is the fix? [1 mark]

- ☐ **A** Raise the look-ahead distance
- ☐ **B** Raise the gain
- ☐ **C** Lower the look-ahead distance
- ☐ **D** Lower the cruise speed to zero on straights

5. At L = 3 and 14 cm/s a follower that steers like a car weaves up to 4.1 cm either side of the tape, but the task's follower, which points its velocity at the look-ahead point, stays within 0.5 cm. Why? [1 mark]

- ☐ **A** Pointing the velocity moves the robot sideways directly, so there is no heading to swing past the line while the lag catches up
- ☐ **B** The task's follower has no drive lag
- ☐ **C** Pointing the velocity uses a longer effective look-ahead
- ☐ **D** The car-like follower drives faster on the straights

6. Why is projecting onto the path and adding L better than intersecting a circle of radius L with the path? [1 mark]

- ☐ **A** It is defined even when the robot is more than L from the path, and it cannot send the robot backwards
- ☐ **B** It gives a shorter look-ahead distance on corners
- ☐ **C** The circle intersection needs the path to be curved
- ☐ **D** It avoids having to compute arc length

7. The lesson's path is built from a straight, a quarter circle as 8 chords, and a straight. What does this print? [1 mark]

```
import math
arc = [(80 + 40 * math.cos(math.radians(180 - k * 11.25)),
        120 + 40 * math.sin(math.radians(180 - k * 11.25))) for k in range(9)]
PATH = [(40.0, 40.0)] + arc + [(150.0, 160.0)]
cum = [0.0]
for (ax, ay), (bx, by) in zip(PATH, PATH[1:]):
    cum.append(cum[-1] + math.hypot(bx - ax, by - ay))
print(round(cum[-1], 1), round(80 + 20 * math.pi + 70, 1))
```

8. Why is the look-ahead usually made to grow with speed, L = L0 + k v? [1 mark]

- ☐ **A** What the controller needs is a roughly constant preview time, and the lag costs a fixed number of seconds
- ☐ **B** A faster robot is further from the path, so it needs a longer reach
- ☐ **C** It keeps the curvature command constant at all speeds
- ☐ **D** It stops the look-ahead point running off the end of the path

The task: chase the look-ahead point

The tape runs straight from (40, 40) to (40, 120), round a quarter circle of radius 40 centred on (80, 120), then straight to (150, 160). Follow it with pure pursuit, plot `off path` and `speed`, print `path:` (the length of the path) and `drove:` (how far the robot actually travelled), and stop at the far end.

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

DT = 0.1
START = (40.0, 40.0)
CRUISE, LOOK = 11.0, 18.0
```

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

Do it on the robot

www.bugbotlab.com/learn/u10-4-pure-pursuit/

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

Challenges

1. Run it at `LOOK = 6` and at `LOOK = 45`. Which one cuts the corner, and by how much? Does the short one weave at all?
2. Steer it like a car instead: drive forwards only, and turn at `v * 2 * sin(alpha) / L` radians a second, where `alpha` is the angle of the look-ahead point from the way the robot faces. Run `L = 3` at 14 cm/s and watch the first straight, then try `L = 5`.
3. Make the look-ahead grow with speed and see whether you can keep both the straights and the corner.
4. Measure the largest `off path` value in the corner and compare it with `L * L / (2 * 40)`.