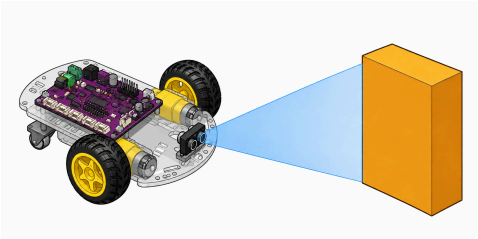
Collision-Avoidance Robot



Build a robot that senses obstacles, stops, backs away, and turns before a collision.

STEPS

1. Mount motors, wheels, and board on the chassis.
2. Wire a forward-facing distance sensor and connect the battery pack.
3. Write drive(left_speed, right_speed): -100 is full reverse; 100 is full forward.
4. Loop: drive forward while the distance exceeds your chosen stop threshold.
5. At the threshold, stop, back up, turn about 90 degrees, then drive forward.
6. Test on a clear floor. Retune for reliable avoidance.

STRETCH GOAL

Try a smaller space. Retune stop distance, speed, backup time, and turn angle.

Advanced Card

Robotics

Solution

HINTS

- *Four things happen in order once an obstacle gets close: stop, reverse, turn, then drive forward again -- one small function per step keeps the loop readable.*
- *A sensor reading that looks perfect on a desk can lie once it's mounted at an angle -- keep it level and pointed straight ahead.*
- *"About 90 degrees" with no encoder really means timing: how many milliseconds you leave the wheels spinning in opposite directions.*

STEPS

1. Confirm your Cytron board's motor pins and your sensor's pins (or bus) from its pinout guide.
2. Write drive(left, right) so a -100-to-100 speed value becomes the right PWM signal on each motor pin.
3. Write get_distance_cm() to return the sensor's live reading in centimeters.
4. Loop forever: drive forward while the reading is above STOP_CM; otherwise reverse, turn, then check the distance again.
5. Start from the chapter's numbers (20 cm, 100 ms) and retune every constant while test-driving in the real room.

```
from machine import Pin, PWM
from time import sleep_ms
LEFT_FWD, LEFT_REV = PWM(Pin(11)), PWM(Pin(10))
RIGHT_FWD, RIGHT_REV = PWM(Pin(8)), PWM(Pin(9))
FULL = 65025
STOP_CM, POLL_MS, REV_MS, TURN_MS = 20, 100, 400, 350

def pwm(speed):
    return int(abs(speed) / 100 * FULL)
def drive(left, right):
    LEFT_FWD.duty_u16(pwm(left) if left > 0 else 0)
    LEFT_REV.duty_u16(pwm(left) if left < 0 else 0)
    RIGHT_FWD.duty_u16(pwm(right) if right > 0 else 0)
    RIGHT_REV.duty_u16(pwm(right) if right < 0 else 0)
def get_distance_cm():
    pass  # read your sensor here

while True:
    if get_distance_cm() > STOP_CM:
        drive(70, 70)
    else:
        drive(-60, -60)
        sleep_ms(REV_MS)
        drive(60, -60)
        sleep_ms(TURN_MS)
    sleep_ms(POLL_MS)
```

See also: Chapter 18's Collision Avoidance Robot section and Control Loop diagram.