

05_01_mapping_room_with_map_teacher

May 2, 2025

Part of the InnovatED STEM and DroneBlocks Land, Air, and Sea Robotics Curriculum
Licensed for educational use in schools only.
Redistribution, commercial use, or resale is strictly prohibited.
© 2025 InnovatED STEM & DroneBlocks. All rights reserved.

1 Mapping a Room with Crazyflie

In this lesson, you'll learn how to use the Crazyflie and its MultiRanger sensors to create a basic 2D map of the environment. We'll collect sensor data while the drone moves, and store it in a data structure for visualization later.

```
[ ]: # Setup for Real Crazyflie
import time
import pandas as pd
from crazyflie_sim import CrazyflieSimulator

drone = CrazyflieSimulator(real=True)
```

1.1 Step 1: Take Off

```
[ ]: drone.takeoff(1.0, 0.3)
time.sleep(2)
```

1.2 Step 2: Initialize Mapping Variables

We'll log the drone's position and sensor values over time into a list.

```
[ ]: # List to store mapping data
room_map = []
```

1.3 Step 3: Move the Drone and Log Sensor Data

We'll move in a simple pattern (e.g., forward & rotate), collecting sensor values at each step.

```
[ ]: # Perform a mapping pass
for _ in range(10):
    data = drone.get_distances()
    position = drone.get_position()
```

```

yaw = drone.get_yaw()

room_map.append({
    'x': position[0],
    'y': position[1],
    'yaw': yaw,
    **data
})

drone.forward(0.2, 0.2)
time.sleep(1)
drone.rotate(30, 0.5)
time.sleep(1)

```

1.4 Step 4: View the Collected Data

```

[ ]: df = pd.DataFrame(room_map)
     df.head()

```

1.5 Exercise 1: Modify the flight path to better cover the room

```

[ ]: # Try zig-zag or spiral paths to map more area
     # Use combinations of forward/left/right/rotate
     # Append position + sensor data to room_map again

```

1.6 Exercise 2: Add a condition to stop mapping if an object is < 0.3m in front

```

[ ]: # if drone.get_distances()['front'] < 0.3:
     #     print("Wall too close! Stopping map.")
     #     break

```

1.7 Step 5: Land and Close

```

[ ]: drone.land(0.3)
     drone.close()

```

1.8 Step 6: Visualize the Room Map

We'll use the drone's position, yaw, and front/left/right/back distances to draw rays (lines) to represent wall detections around the room.

```

[ ]: import math
     import matplotlib.pyplot as plt

     fig, ax = plt.subplots(figsize=(6, 6))
     ax.set_title("Mapped Room View (Top Down)")

```

```

ax.set_xlabel("X (m)")
ax.set_ylabel("Y (m)")

# Draw each sensor ray
for point in room_map:
    x, y, yaw = point['x'], point['y'], math.radians(point['yaw'])

    for direction, angle_offset in [('front', 0), ('left', math.pi/2),
    ↪ ('right', -math.pi/2), ('back', math.pi)]:
        dist = point[direction]
        if isinstance(dist, (float, int)) and dist < 2.0:
            angle = yaw + angle_offset
            end_x = x + dist * math.cos(angle)
            end_y = y + dist * math.sin(angle)
            ax.plot([x, end_x], [y, end_y], color='red', alpha=0.3)

# Draw drone path
xs = [p['x'] for p in room_map]
ys = [p['y'] for p in room_map]
ax.plot(xs, ys, 'bo-', label='Drone Path')
ax.legend()
ax.grid(True)
ax.axis('equal')
plt.show()

```