

BEV2D栅格占用建图

Multisensor Fusion 课程汇报

BEV2D栅格占用建图

刘商鉴 2022310962

2024.12.22

1. 本文的工作

本文复现了课程topic4利用多向超声传感器进行BEV2D栅格占用建图的早期工作。这项工作名为Sonar-based real-world mapping and navigation。由CMU的Alberto Elfes. 发表在IEEE Journal of Robotics and Automation 3, 249-264 (1987).

1.1 论文的大致内容

声纳系统能够通过发射声波并分析返回信号来获取周围环境的信息。这种技术特别适用于复杂和动态的环境，因为它可以在不同的条件下提供可靠的距离测量（如低光照或复杂背景）。文章的目标是提出一种基于声纳的映射和导航方法，能够在动态环境中实现高效的路径规划和障碍物识别。

1.2 论文包含多个层级的内容：

VIII. Supervisor

- Global Supervision of System Behaviour
- User Interface

VII. Global Planning

- Task-Level Planning to provide sequences of sensory, actuator and processing (software) actions
- Simulation
- Error Recovery and Replanning in case of failure or unexpected events

VI. Control

- Scheduling of Activities
- Integration of Plan-Driven with Data-Driven Activities

V. Navigation

- Navigation Modules provide services such as Path-Planning and Obstacle Avoidance

IV. Real-World Modelling

- Integration of local pieces of correlated information into a Global Real-World Model that describes the robot's environment of operation
- Matching acquired information against stored maps

- Object Identification
- Landmark Recognition

III. Sensor Integration

- Information provided by different Sensor Modules is correlated and abstracted
- Common representations and compatible frames of reference are used

II. Sensor Interpretation

- Acquisition of Sensor Data (Vision, Sonar, Laser Rangefinder, etc.)
- Interpretation of Sensor Data

I. Robot Control

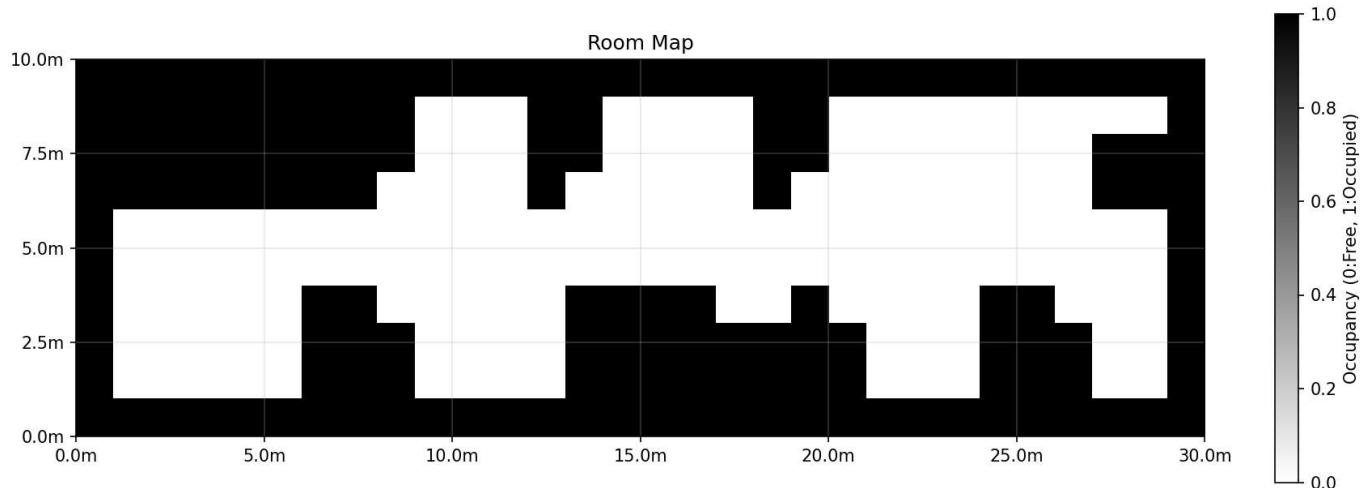
- Set of Primitives for Robot Operation
- Actuator Control (e.g., Locomotion)
- Sensor Control
- Internal Sensor Monitoring

我们的复现也从传感器以及基本的机器人建模开始。完成机器人建模之后对多次测量数据进行融合。最后将建立的机器人和数据融合算法应用于BEV建图和导航。

2 复现过程

2.1 环境地图生成

在生成环境地图过程中，我们首先生成环境地图的边界。然后在边界上以及房间中生成随机的物品以模拟房间内物品的摆放。最后通过重采样得到预期分辨率的地图。



代码地址：https://github.com/liushangjian/BEV2D-simulation/blob/main/map_generator.py

2.2 单个传感器建模

超声传感器有以下一些特征需要在建模中体现：

- 超声波是扇形波，每次探测一个扇形范围
- 随着声波传播其强度逐渐衰减，可信度下降

- 超声波检测对距离敏感，但是对角度不敏感

适用于超声波传感器的概率分布：

参数定义

- $P = (x, y, z)$: 属于声纳波束所扫过的体积中的一个点。
- R : 声纳传感器返回的范围测量值。
- E : 最大声纳测量误差。
- w : 传感器波束宽度。
- D : 主要波瓣的灵敏度所占的立体角。
- $S = (x_s, y_s, z_s)$: 声纳传感器的位置。
- δ : 从点P到S的距离。
- θ : 从S看P时波束主轴与P之间的角度。

概率密度函数

1. Probably Empty Region:

- 包括声纳波束内的点 ($\delta < R - E$ 和 $\theta \in [-\frac{\pi}{2}, \frac{\pi}{2}]$)，其为空的概率为：

$$p_E(x, y, z) = P[\text{point } (x, y, z) \text{ is empty}] = f_E(\delta, \theta)$$

- 其中：

$$E_r(\delta) = 1 - \left(\frac{(\delta - R_{min})}{(R - E - R_d)} \right)^2, \quad \text{for } \delta \in [R_{min}, R - E]$$

$$E_r(\delta) = 0, \quad \text{otherwise}$$

$$E_a(\theta) = 1 - \left(\frac{2\theta}{w} \right)^2, \quad \text{for } \theta \in [-\frac{w}{2}, \frac{w}{2}]$$

2. Occupied Probability Density Function:

- 对于声纳波束前面的点P，其被占用的概率为：

$$p_o(x, y, z) = P[\text{position } (x, y, z) \text{ is occupied}] = O_r(\delta) \cdot O_a(\theta)$$

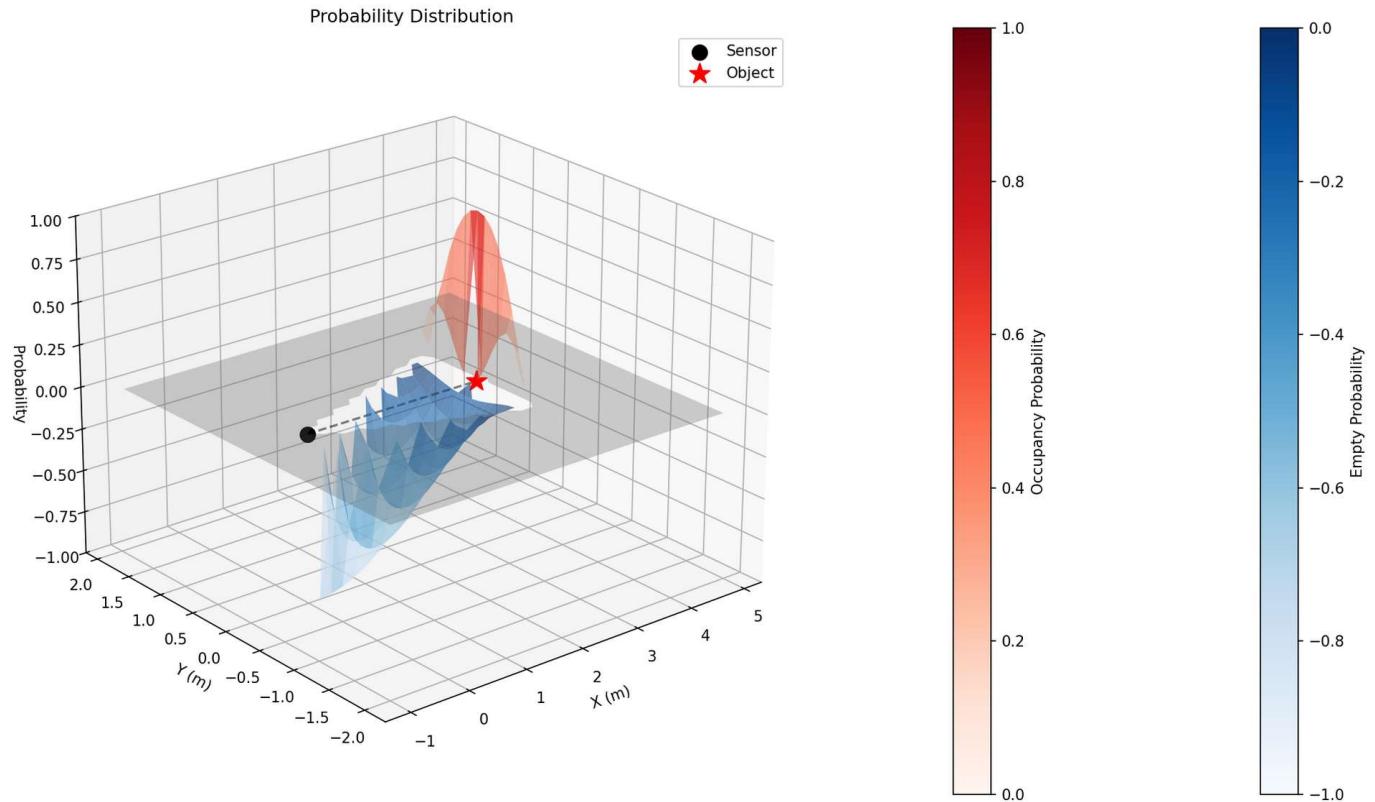
- 其中：

$$O_r(\delta) = 1 - \left(\frac{(\delta - R)}{w} \right)^2, \quad \text{for } \delta \in [R - E, R + E]$$

$$O_r(\delta) = 0, \quad \text{otherwise}$$

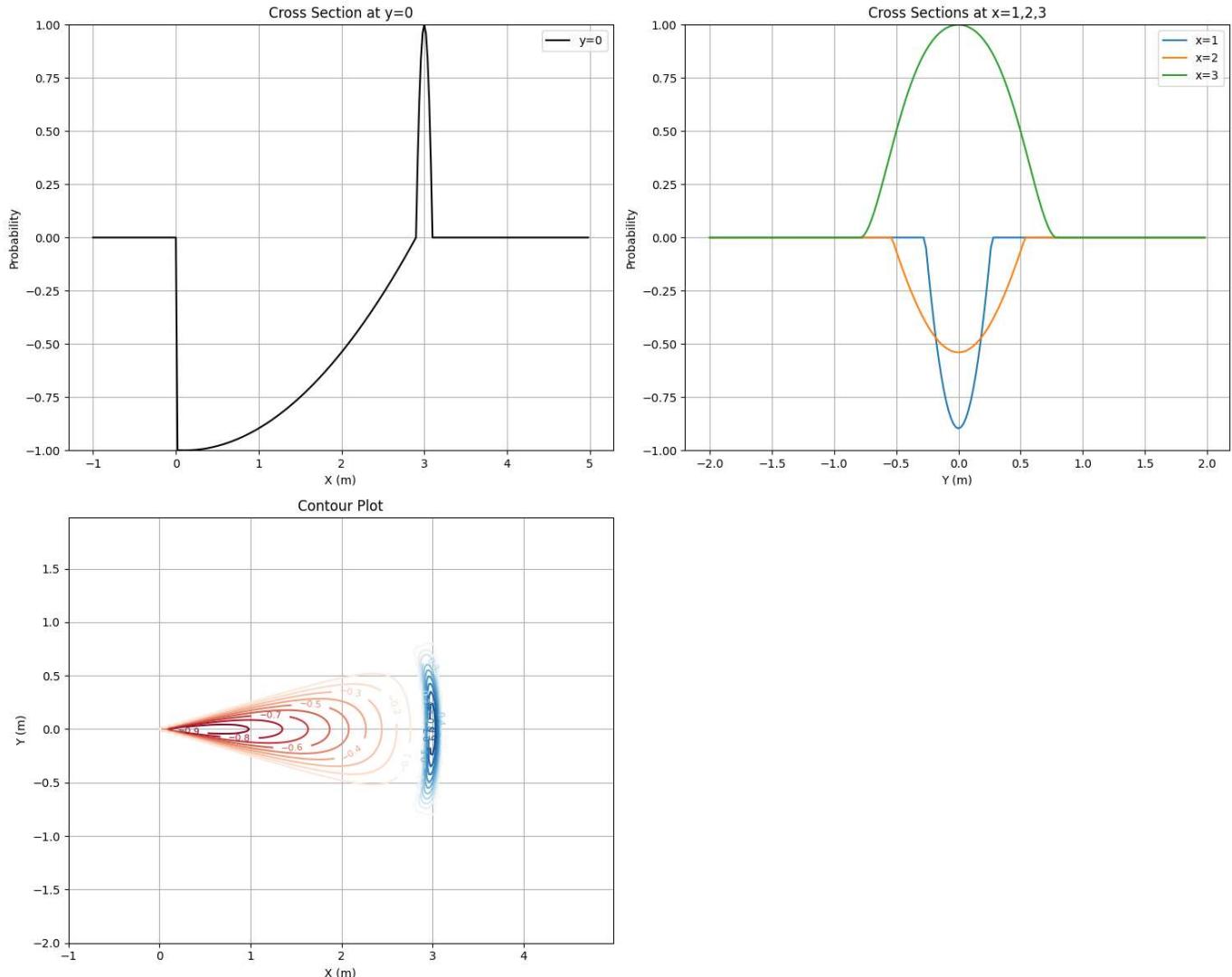
$$O_a(\theta) = 1 - \left(\frac{2\theta}{w} \right)^2, \quad \text{for } \theta \in [-\frac{w}{2}, \frac{w}{2}]$$

对这一概率进行绘图：



将概率分为2支进行处理，一支是空置概率，另一支是占据概率。从传感器到物体之前的区域被划为空置区。而物体位置则被定义为占据区。

对这一概率密度函数进行截取：



代码位置：https://github.com/liushangjian/BEV2D-simulation/blob/main/single_sonar_simulation.py

2.3 机器人建模

- 1, 模拟装备有8个超声波传感器的机器人。在单个传感器的基础上，模拟8个传感器，其传感域不重叠。机器人具有位置，角度两个参数。涉及到机器人和世界坐标的变换。
- 2, 定义了扇形检测算法：在超声范围内发出多个射线，计算射线焦点，计算射线焦点到传感器的距离来作为检测结果。
- 3, 机器人每一次传感输出 $10m \times 10m$ 的二维地图，地图上每个点有8个传感器的结果。机器人输出的地图可以进行拼接或者裁剪来整合到世界地图上。
未来也可能有些没有地图限制的脚步

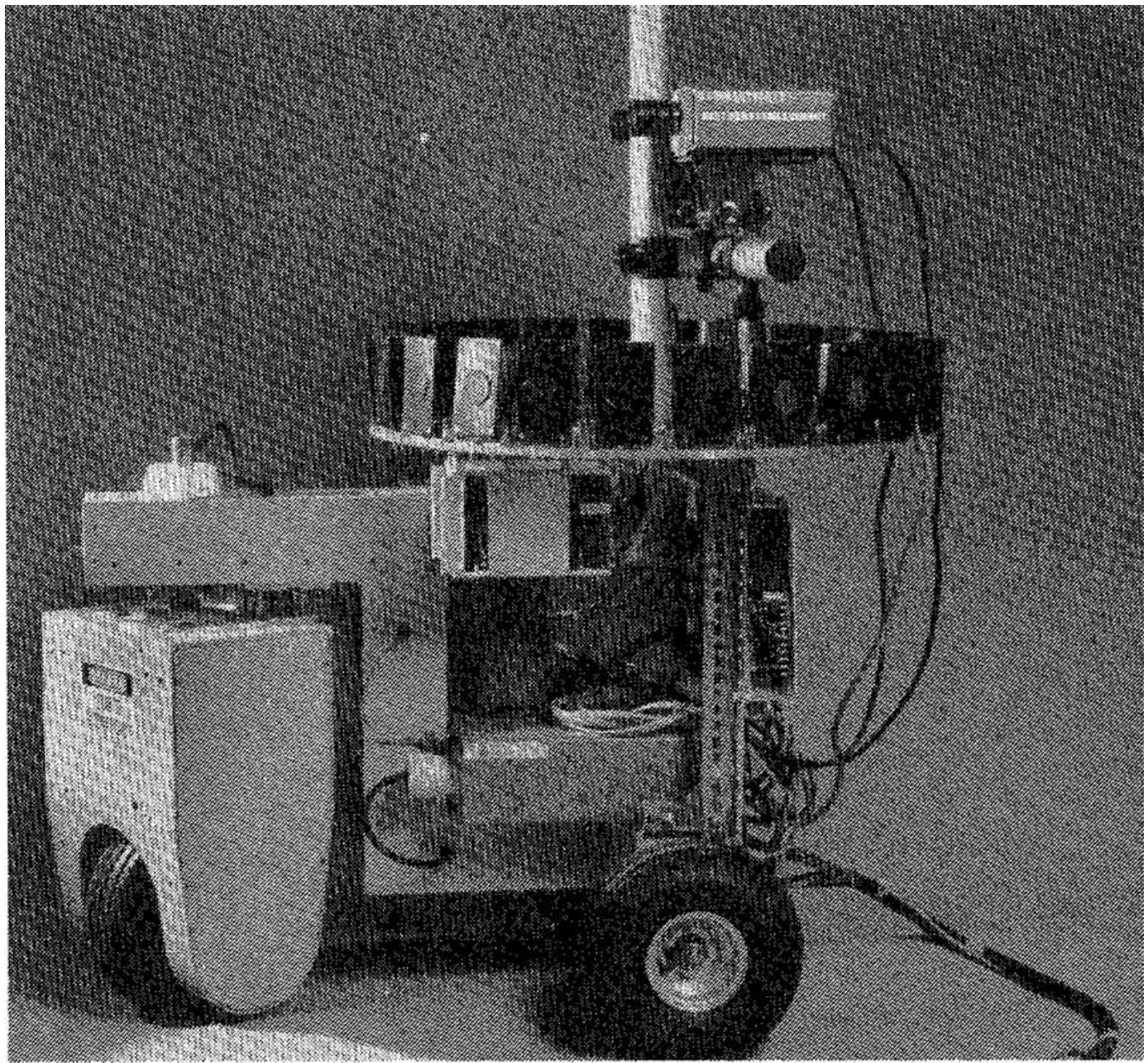


Fig. 2. Neptune mobile robot, with pair of cameras and sonar ring.

文中的机器人：

机器人完成一次建图模拟：

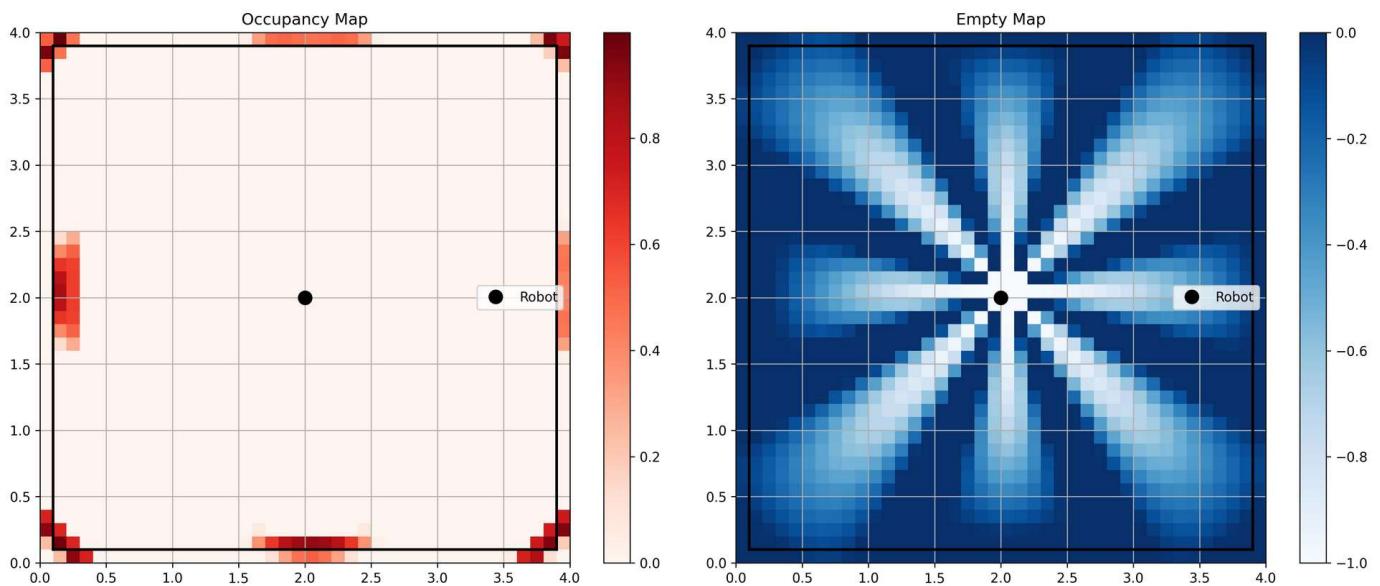
具有8个方向的超声传感机器人完成室内建图的工作。先假设是一个 $[4, 4]m$ 的房间，房间四周有 $0.1m$ 厚的墙（墙在房间内）。机器人在房间中心。机器人朝向X轴正方向。

机器人完成一次测量，绘制出其occupacymap和empty map。map的表示范围：

unknown: 0

empty: $(-1, 0)$

occupied: $(0, 1]$



机器人完成一次建图的模拟图：

代码位置：https://github.com/liushangjian/BEV2D-simulation/blob/main/eight_sonars_robot_single_mapping.py

2.4 多次读数之间的融合

融合逻辑是fuzzy logic, 这个逻辑对于occupancy map和empty map的融合是不对称的。因为empty的确定度较高，如果没检测到大概率就是真的没有，所以用加法逻辑。而occupancy的确定度较低，如果检测到，也只是说明那一个区域有东西，不代表具体某个点有东西。所以其概率可以被empty削弱。

具体的融合逻辑如下：

Superposition of empty areas: The probabilities corresponding to empty areas are combined using the probability addition formula:

$$\$ p_E(\text{cell}) = p_E(\text{cell}) + p_E(\text{reading}) - p_E(\text{cell}) \times p_E(\text{reading}). \$$$

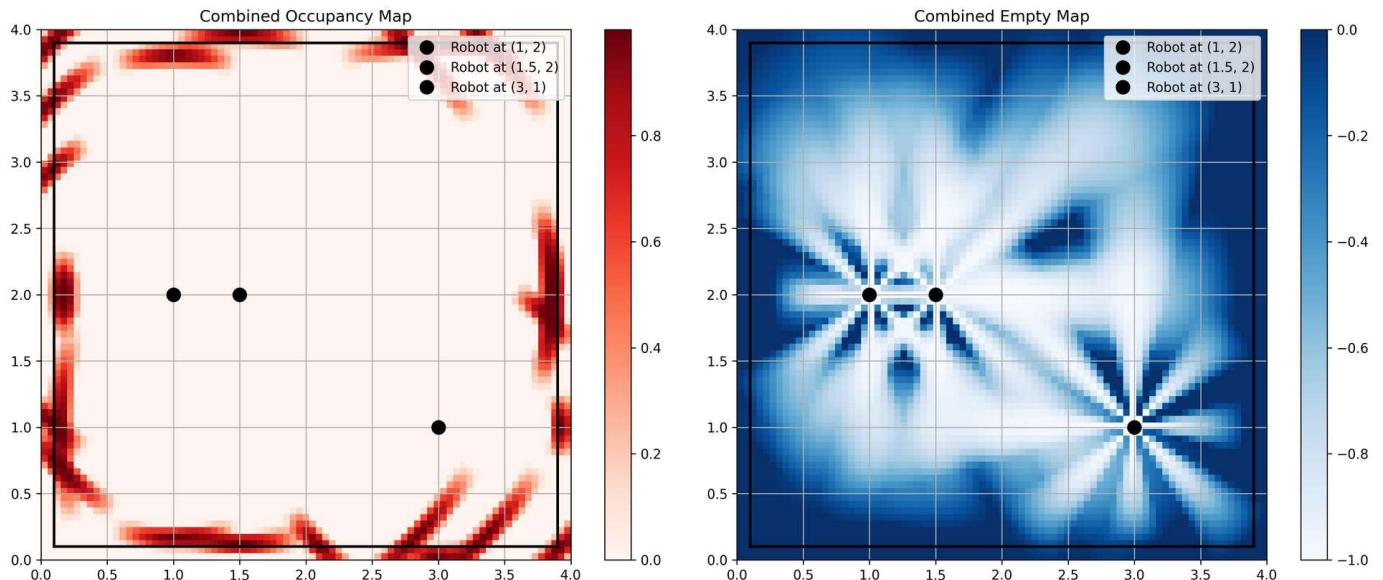
Superposition of occupied areas: The probabilities corresponding to occupied areas are initially weakened by the evidence of the empty certainty factors, using:

$$p_O(\text{reading}) = p_O(\text{reading}) \cdot (1 - p_E(\text{cell})).$$

We then normalize the occupied probabilities over the beam front. Finally, the occupied probabilities are combined using:

$$p_O(\text{cell}) = p_O(\text{cell}) + p_O(\text{reading}) - p_O(\text{cell}) \times p_O(\text{reading}).$$

三个传感器测量融合之后的结果：



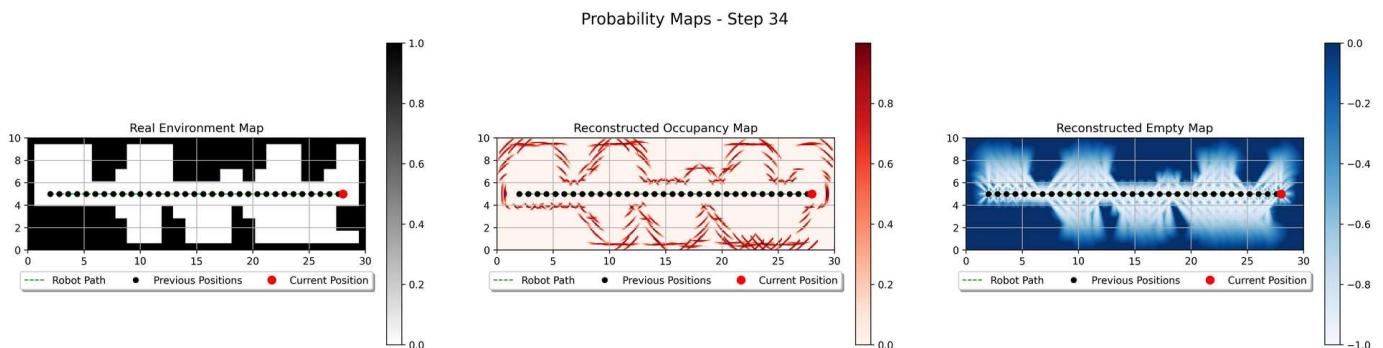
代码地址：https://github.com/liushangjian/BEV2D-simulation/blob/main/multi_position_mapping.py

2.5 对地图进行建图融合

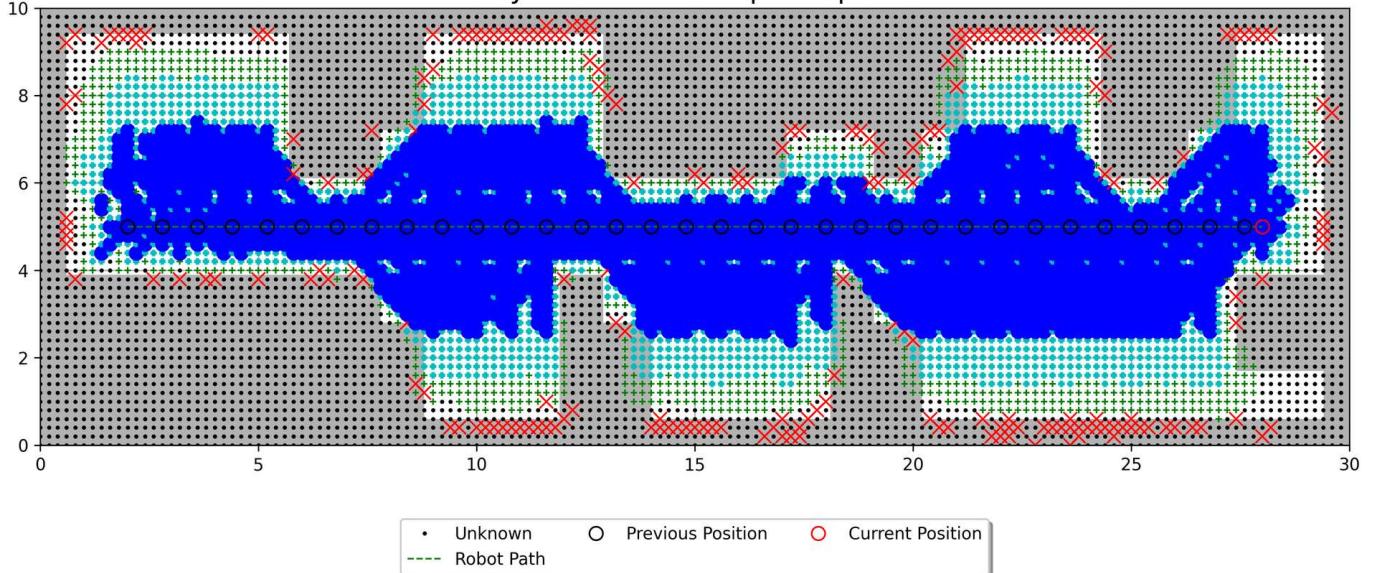
这个程序模拟了机器人沿着带有随机偏移的直线路径进行环境探测的过程，通过多次测量逐步建立环境地图，并保存每一步的探测结果。整个过程是渐进式的，可以观察到地图是如何随着机器人移动逐步构建的。

整体来说，对室内的建图与之前的多传感器融合没有太多差别，只是换了一个更大的地图，在技术上也没有很多需要更新的地方。

直线路径建图的结果：

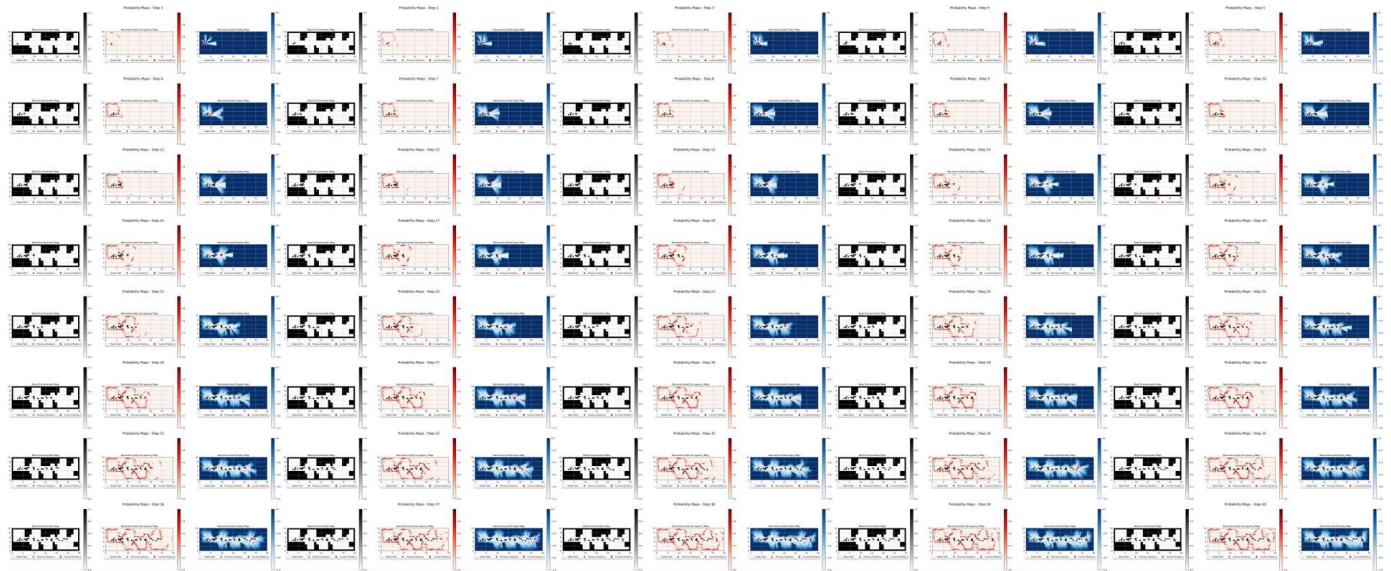


Symbolic Sonar Map - Step 34



能够看到房间的轮廓被较好的标记为占用。而房间空置的位置则被标记为empty。但是因为超声的取向一致，所以有一些周期性无法探测到的地方。因此引入随机移动建模，使得机器人能够以不同的取向进行检测，消除无法检测到的区域。

随机路径建图的过程：



随机采用的是在原有的直线路线上随机化步长并添加噪声实现的。

```
# 计算方向向量
direction = np.array(end) - np.array(start)
distance = np.linalg.norm(direction)
unit_direction = direction / distance

# 生成随机距离 (均匀分布)
random_distances = np.random.uniform(0, distance, num_points)
random_distances.sort() # 按距离排序，确保机器人从近到远移动

# 生成y轴随机误差
```

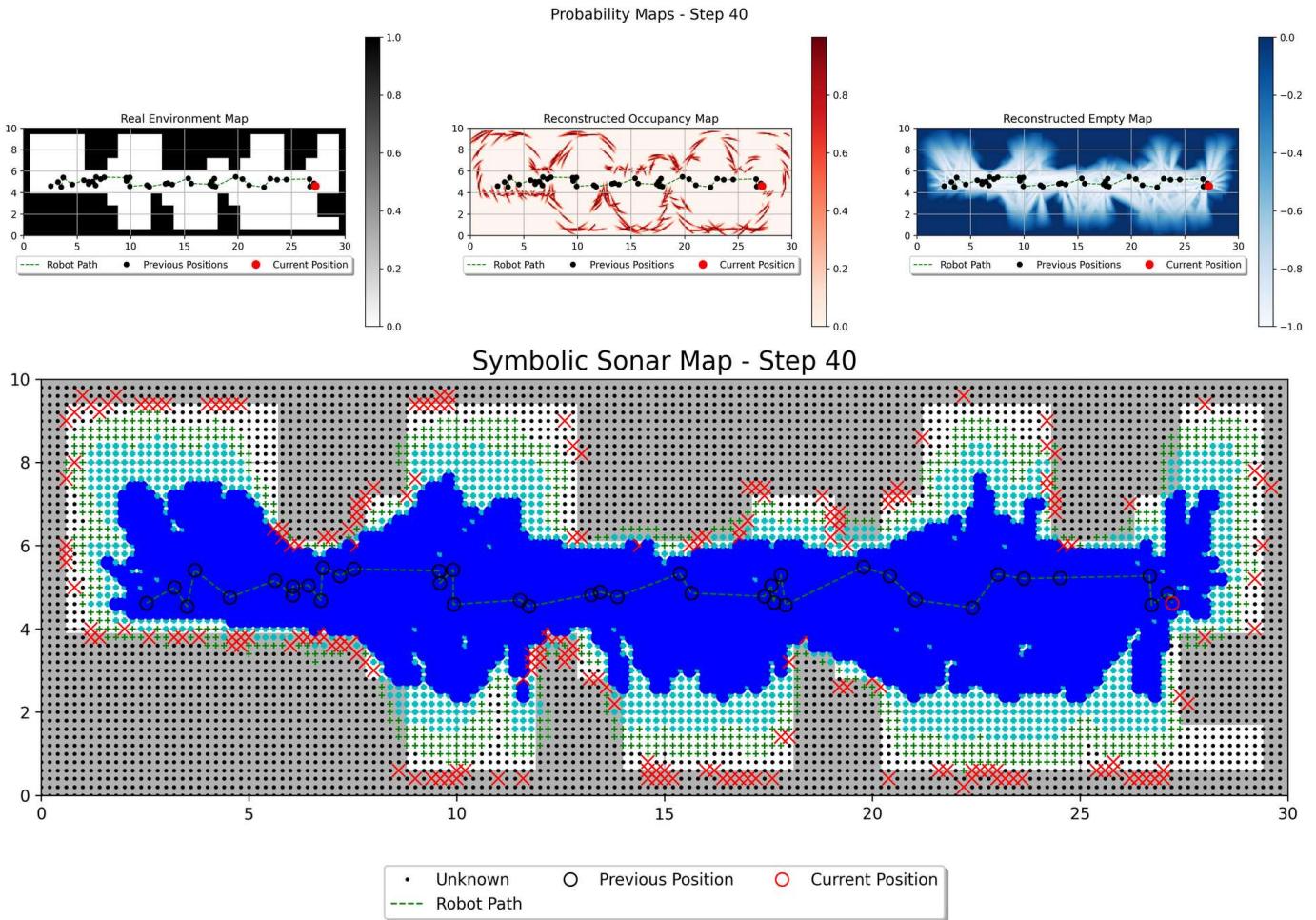
```

y_errors = np.random.uniform(-y_noise, y_noise, num_points)

# 生成测量位置
positions = []
for t, y_err in zip(random_distances, y_errors):
    # 计算基础位置
    base_position = np.array(start) + t * unit_direction
    # 添加y轴误差
    position = base_position + np.array([0, y_err])
    positions.append(tuple(position))

```

随机建图的结果：



随机化移动之后有效的消除了直线运动时周期性的检测盲点，使得地图核心区域更加完善。

重建地图采用的是阈值法，首先判断是否为占据，然后判断是否为空置：

```

if occupancy > 0.7:  # 被占用区域
    ax4.plot(x, y, 'rx', markersize=8, label='Occupied' if i==0 and j==0 else "")
elif empty > 0.7:  # 高确定性空区域
    ax4.plot(x, y, 'b.', markersize=16, label='Empty (High Certainty)' if i==0 and j==0 else "")
elif empty > 0.4:  # 中等确定性空区域
    ax4.plot(x, y, 'c+', markersize=4, markeredgewidth=2, label='Empty (Medium Certainty)' if i==0 and j==0 else "")

```

```

Certainty)' if i==0 and j==0 else ""))
elif empty > 0.2: # 低确定性空区域
ax4.plot(x, y, 'g+', markersize=4, markeredgewidth=1, label='Empty (Low Certainty)')
if i==0 and j==0 else ""))
else: # 未知区域
ax4.plot(x, y, 'k.', markersize=3, label='Unknown' if i==0 and j==0 else "")

```

得到了和论文中质量接近的重建结果：

原文重建结果：

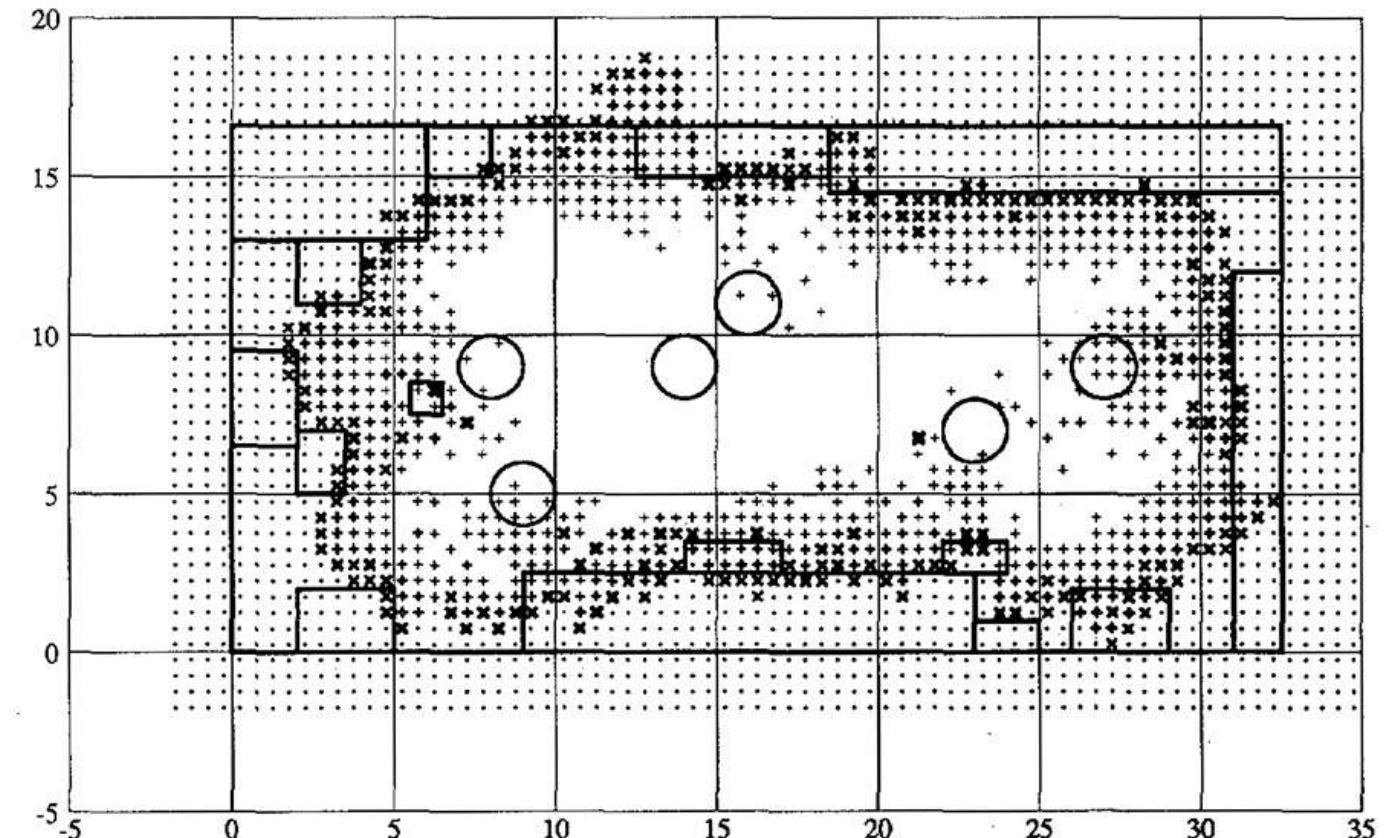


Fig. 8. Two-dimensional sonar map after thresholding.

代码位置：

https://github.com/liushangjian/BEV2D-simulation/blob/main/linear_random_path_mapping.py

https://github.com/liushangjian/BEV2D-simulation/blob/main/linear_path_mapping.py

2.6 机器人自主导航系统

根据文章中的架构实现了导航功能。基于传感器数据融合产生的概率地图进行导航。能够有效避免和房间内实物的碰撞。

整体控制流程：

机器人在每个位置进行环境感知

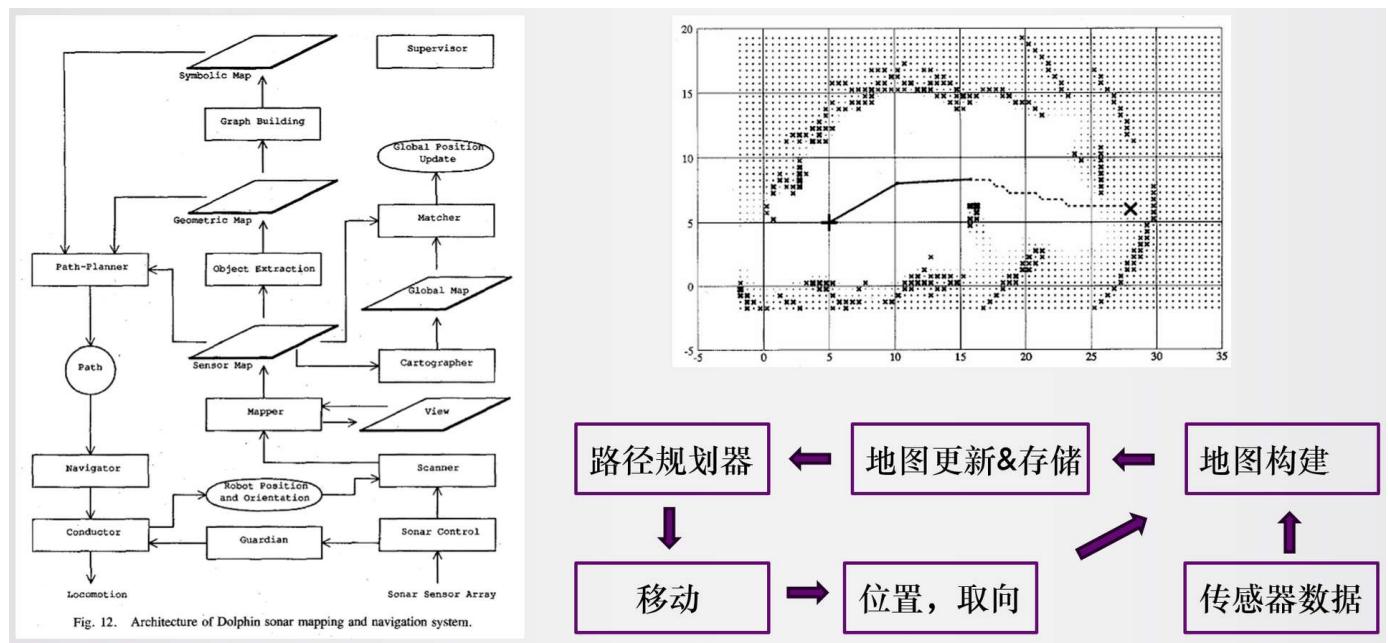
更新全局概率地图

基于当前地图规划下一步移动

选择最优移动方向并执行

记录路径并保存状态
重复以上步骤直到达到目标

导航系统架构:



1. 传感器系统 (RobotWithEightSonars)

- 八个超声波传感器，均匀分布在机器人周围（每45度一个）
- 每个传感器的测量范围：0.2m - 5.0m
- 传感器模拟了真实的测量误差和噪声
- 通过 `simulate_measurements` 方法获取周围环境的距离信息

2. 概率地图构建 (MultiPositionMapping)

地图系统

- 使用两个概率地图：
`occupancy_map`: 记录障碍物位置
`empty_map`: 记录空闲区域
- 地图更新逻辑
对每个传感器测量值，计算光线路径
更新光线路径上的空闲概率
更新测量终点的障碍物概率

多位置融合

保存全局地图
将新的局部观测整合到全局地图中

3. 自主导航控制 (NavigationRobot)

- 初始化
起始位置: (2.5, 7.0)
目标位置: (29.0, 1.0)

- 导航循环
当距离目标 $> 0.5m$ 时:
 - a. 环境感知
使用八个超声波传感器获取周围环境信息
更新概率地图 (障碍物和空闲区域)
 - b. 路径规划
检查二级相邻格子 (步长0.4m)
对直线和斜向移动使用不同步长:
 - 直线移动: 完整步长
 - 斜向移动: 步长/2通过检查中点和终点确保路径安全
 - c. 移动决策
从所有安全路径中选择最接近目标的方向
更新机器人位置和朝向
 - d. 状态保存
记录路径历史
保存地图状态和可视化结果

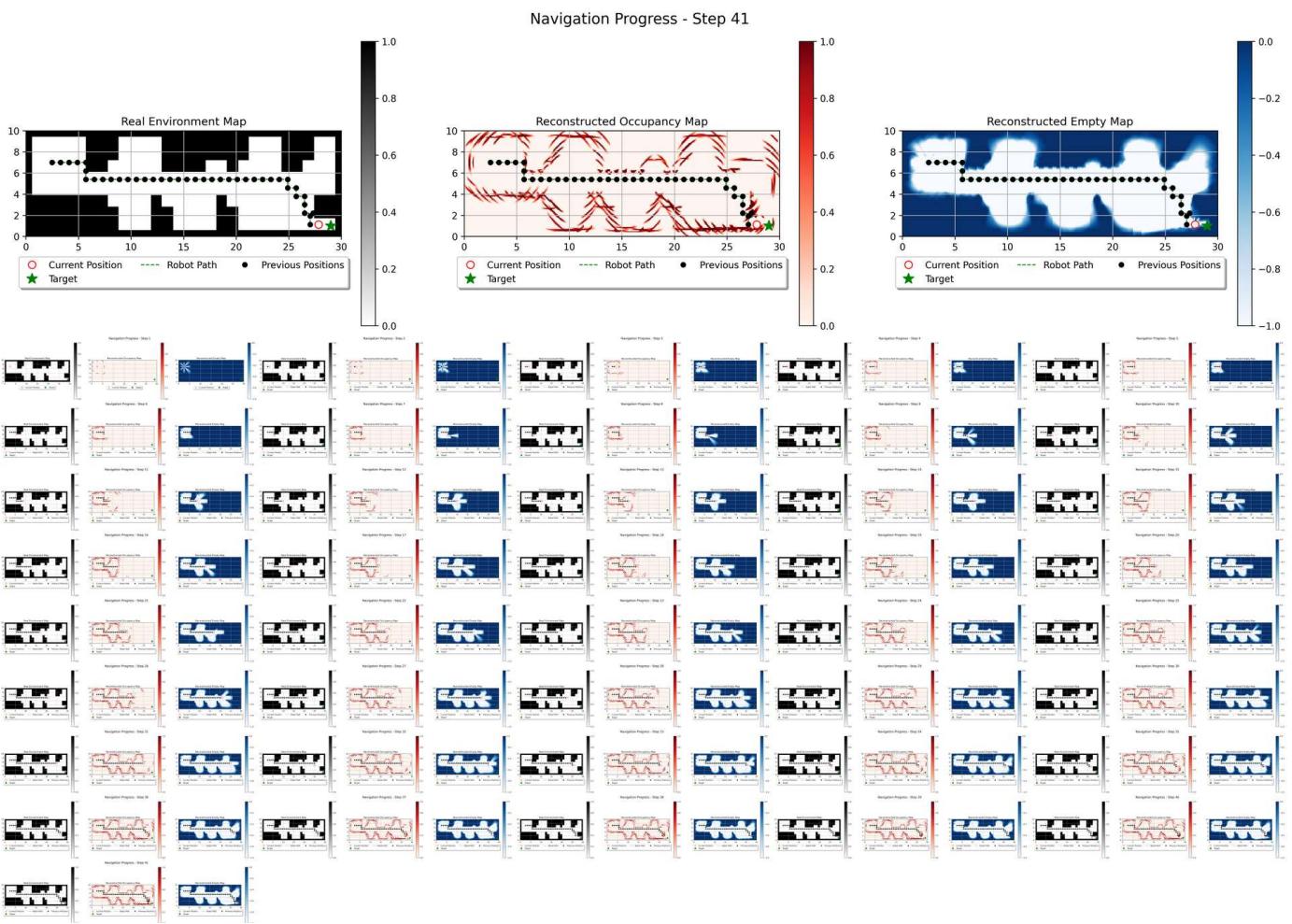
4. 安全性检查机制

检查点必须在地图范围内 (0-30m, 0-10m)
占用概率 < 0.5
空置概率 $<= -0.2$

5. 地图坐标系统

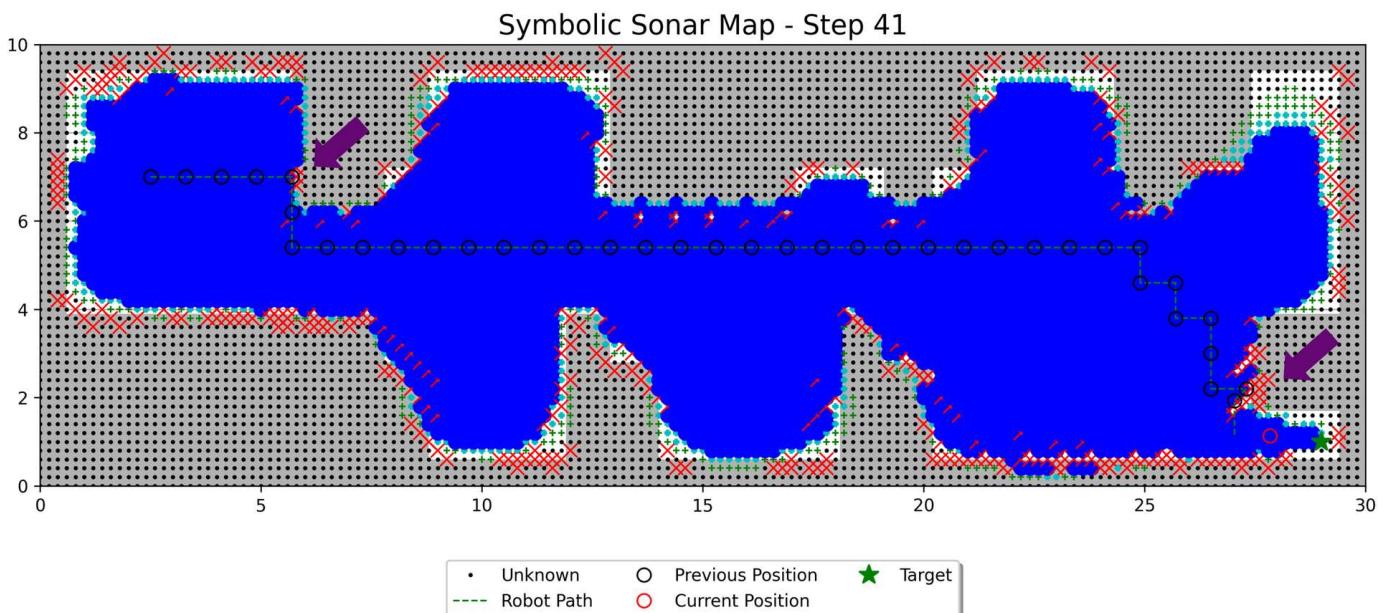
网格分辨率: 0.1m
在每一次机器人传回数据时进行更新。

机器人自动导航时的位置和探索过程:
出发点位于左下角, 终点位于右下角。

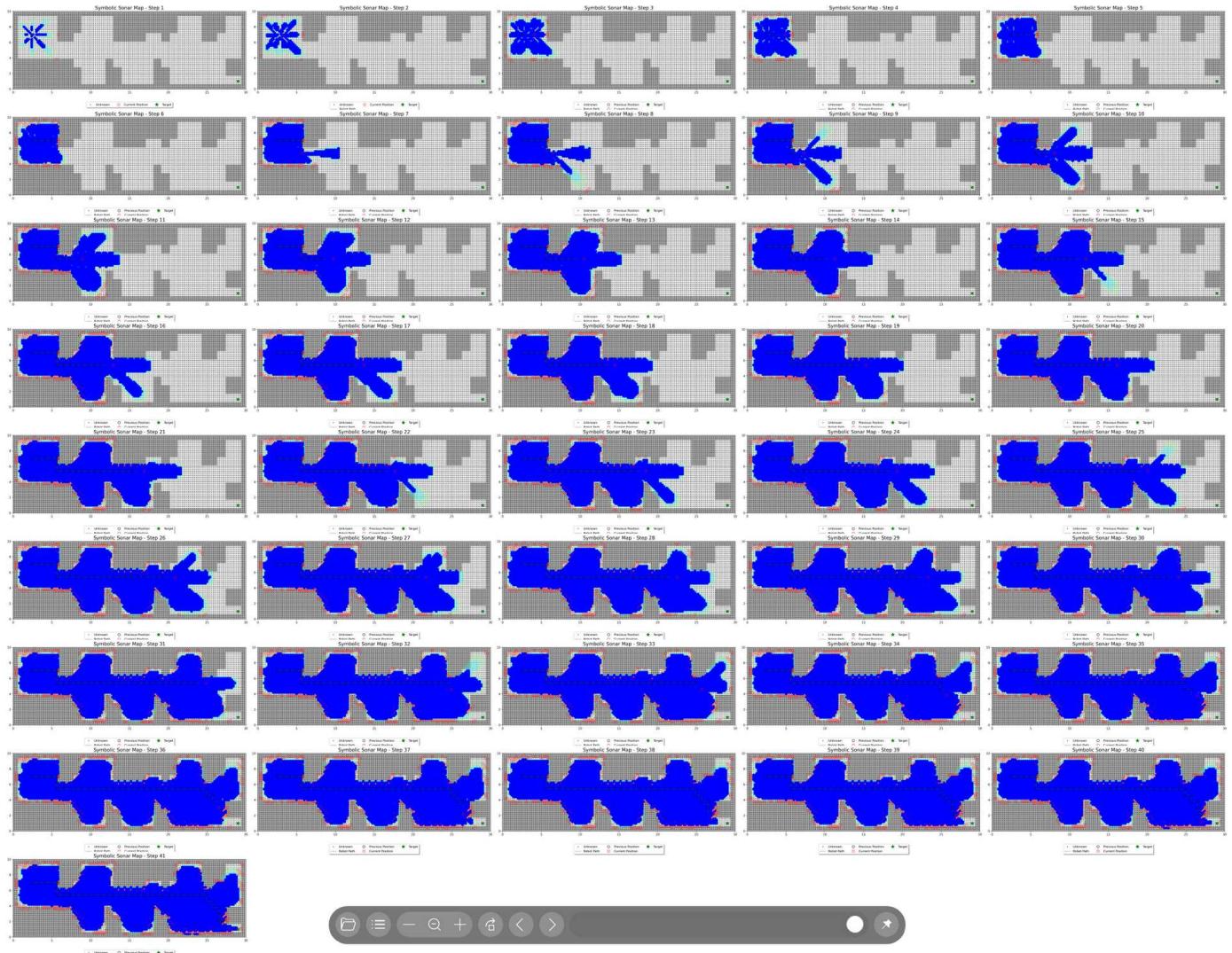


目前的代码没有使用搜索算法，而是采用已知终点最速下降法。并在这个基础上做了一定的优化使其能够避开较小的障碍。未来可能需要采用搜索算法来增强未知环境下找寻物体的能力。

导航机器人行动路径及其重建结果（紫色箭头展示其躲避障碍物行为）：



导航过程中的地图构建：



代码位置：

https://github.com/liushangjian/BEV2D-simulation/blob/main/autonomous_navigation.py.

参考文献

Alberto Elfes. Sonar-based real-world mapping and navigation. IEEE Journal of Robotics and Automation 3, 249-264 (1987).

A. B. Bagheri, Sonar Signal Processing, in Applications of Digital Signal Processing, Signal Processing Series. Englewood Cliff, NJ: Prentice-Hall, 1978.