智能机器人作业

一、使用说明

1. **安装 Pygame：** 确保你已经安装了 Pygame。如果没有安装，可以在终端或命令提示符中执行以下命令：pip install pygame
2. **运行代码**
3. **绘制墙壁和设置目标点：**
   - 使用鼠标左键在窗口中绘制墙壁。
   - 使用鼠标右键设置目标点。
4. **选择算法：** 按任意键开始寻路算法。在开始算法前，确保已经设置了起点和目标点。
5. **选择算法类型：** 在终端或命令提示符中，程序会要求你输入 1（Dijkstra）或 2（A*）来选择使用广度优先搜索或A*算法。
6. **查看结果：** 程序会在图形界面中显示迷宫、寻路过程和找到的路径。如果没有找到路径，会弹出消息框提示无解。

二、用到的算法

dijkstra、A*算法

> dijkstra

Dijkstra算法是一种用于解决带有非负权重边的图的单源最短路径问题的算法。它由荷兰计算机科学家 Edsger Dijkstra 在1956年提出，是最早的路径规划算法之一。Dijkstra算法保证对于所有节点v，它找到的从源节点到v的路径是最短的。

算法步骤：

1. **初始化：** 将起始节点的最短路径估计设为0，其他节点的最短路径估计设为无穷大（或一个足够大的值）。
2. **遍历：** 从起始节点开始，遍历图中的所有节点。
3. **选择节点：** 选择未标记的节点中最短路径估计最小的节点作为当前节点。
4. **更新路径估计：** 对于当前节点的每个邻居节点，计算从起始节点经过当前节点到达邻居节点的路径长度。如果这条路径的长度小于邻居节点当前的最短路径估计，则更新邻居节点的最短路径估计。

5. **标记节点：** 将当前节点标记为已访问。
6. **重复：** 重复步骤3到步骤5，直到所有节点都被标记为已访问，或者目标节点被标记为已访问。
7. **路径重建：** 如果找到目标节点，通过回溯从目标节点到起始节点的路径。

> A*算法

A*（A星）算法是一种启发式搜索算法，常用于图的路径规划和图搜索问题。

算法步骤：

1. **初始化：** 将起始节点放入开放列表（Open List）中，并将其估计代价设为0。
2. **迭代：** 从开放列表中选择估计代价最小的节点作为当前节点，将其移入封闭列表（Closed List），并检查是否达到目标节点。
   - 如果是目标节点，算法结束。
   - 否则，扩展当前节点的邻居节点，计算它们的估计代价（启发式值 + 已走路径的实际代价），将它们放入开放列表中。
3. **更新代价：** 如果已经有更低的估计代价到达某个节点，更新该节点的估计代价和路径。
4. **重复：** 重复步骤2和步骤3，直到找到目标节点或开放列表为空。
5. **路径重建：** 如果找到目标节点，通过回溯从目标节点到起始节点的路径。

三、说明

本程序基于maxontech/dijkstra-pathfinding: Visualization 👓 of Dijkstra's pathfinding algorithm (github.com)开发

在原有基础上增加了A*算法