jupyter.org/try-jupyter/notebooks/notebooks%2F%20A*search.ipynb

jupyterlite A*search Last Checkpoint: 4 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab

Python (Pyodide)

[1]:
```
import heapq

class Node:
    def __init__(self, position, g, h):
        self.position = position
        self.g = g  # Cost from start to current node
        self.h = h  # Heuristic (estimated cost to goal)
        self.f = g + h  # Total cost
        self.parent = None

    def __lt__(self, other):
        return self.f < other.f

def heuristic(a, b):
    """Compute Manhattan distance heuristic."""
    return abs(a[0] - b[0]) + abs(a[1] - b[1])

def astar(grid, start, goal):
    open_list = []
    closed_set = set()

    start_node = Node(start, 0, heuristic(start, goal))
    heapq.heappush(open_list, start_node)
```

jupyter.org/try-jupyter/notebooks/notebooks%2F%20A*search.ipynb

jupyterlite A*search Last Checkpoint: 4 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python (Pyodide)

```
while open_list:
    current_node = heapq.heappop(open_list)

    if current_node.position == goal:
        path = []
        while current_node:
            path.append(current_node.position)
            current_node = current_node.parent
        return path[::-1]  # Reverse to get correct order

    closed_set.add(current_node.position)

    for dx, dy in [(-1,0), (1,0), (0,-1), (0,1)]:  # 4-way movement
        neighbor_pos = (current_node.position[0] + dx, current_node.position[1] + dy)

        if (0 <= neighbor_pos[0] < len(grid)) and (0 <= neighbor_pos[1] < len(grid[0])) and grid[neighbor_pos[0]][neighbor_pos[1]] == 0:
            if neighbor_pos in closed_set:
                continue

            g = current_node.g + 1
            h = heuristic(neighbor_pos, goal)
            neighbor_node = Node(neighbor_pos, g, h)
            neighbor_node.parent = current_node

            heapq.heappush(open_list, neighbor_node)
```

jupyter.org/try-jupyter/notebooks/notebooks%2F%20A*search.ipynb

jupyterlite A*search Last Checkpoint: 4 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python (Pyodide)

```
                g = current_node.g + 1
                h = heuristic(neighbor_pos, goal)
                neighbor_node = Node(neighbor_pos, g, h)
                neighbor_node.parent = current_node

                heapq.heappush(open_list, neighbor_node)

    return None  # No path found

# Example grid (0 = walkable, 1 = obstacle)
grid = [
    [0, 0, 0, 0, 1],
    [1, 1, 0, 1, 0],
    [0, 0, 0, 1, 0],
    [0, 1, 1, 0, 0],
    [0, 0, 0, 0, 0]
]

start = (0, 0)
goal = (4, 4)

path = astar(grid, start, goal)
print("optimal Path:", path if path else "No path found")
```

```
Shortest Path: [(0, 0), (0, 1), (0, 2), (1, 2), (2, 2), (2, 1), (2, 0), (3, 0), (4, 0), (4, 1), (4, 2), (4, 3), (4, 4)]
```