**Date:22.10.25**

**TASK:11**

**Implementation of ROBOT TRAVERSAL Pathfinding.**

**CO1, CO2, CO3 S3**

**PROBLEM STATEMENT  
Design and implement a program to simulate a robot traversing a 2D grid with obstacles, finding an optimal path from a starting point to a destination.**

**AIM  
To develop a simulation that demonstrates how a robot can autonomously navigate a grid, avoiding obstacles using an efficient traversal algorithm.**

**OBJECTIVE**

* **Model the environment as a 2D grid with marked obstacles.**
* **Allow user input for robot start and end positions.**
* **Implement a traversal algorithm (e.g., A\* or BFS) for pathfinding.**
* **Output the traversed path and steps taken.**

**DESCRIPTION  
This use case involves creating a virtual environment where a robot moves from a designated starting cell to a target cell in a grid with randomly placed obstacles. A pathfinding algorithm determines the shortest available route, with the robot’s path displayed visually or textually.**

**ALGORITHM**

1. **Initialize the grid and place obstacles.**
2. **Set robot's starting and ending positions.**
3. **Apply the chosen traversal algorithm (such as A\*) to explore possible paths.**
4. **Mark the cells traversed by the robot.**
5. **Stop when the destination is reached or declare failure if unreachable.**

**PROGRAM  
import heapq**

**def heuristic(a, b):**

**# Manhattan distance heuristic**

**return abs(a[0] - b[0]) + abs(a[1] - b[1])**

**def astar(grid, start, end):**

**rows, cols = len(grid), len(grid[0])**

**queue = []**

**heapq.heappush(queue, (0 + heuristic(start, end), 0, start, [start]))**

**visited = set()**

**while queue:**

**(est\_total\_cost, cost, node, path) = heapq.heappop(queue)**

**if node == end:**

**return path**

**if node in visited:**

**continue**

**visited.add(node)**

**for dx, dy in [(-1,0), (1,0), (0,-1), (0,1)]: # move up, down, left, right**

**nx, ny = node[0]+dx, node[1]+dy**

**if 0 <= nx < rows and 0 <= ny < cols:**

**if grid[nx][ny] == 0 and (nx, ny) not in visited:**

**heapq.heappush(queue, (cost+1+heuristic((nx, ny), end), cost+1, (nx, ny), path+[(nx, ny)]))**

**return None**

**# Example usage**

**N, M = 5, 5**

**grid = [[0, 1, 0, 0, 0], # 0: free cell, 1: obstacle**

**[0, 1, 0, 1, 0],**

**[0, 0, 0, 1, 0],**

**[1, 1, 0, 1, 0],**

**[0, 0, 0, 0, 0]]**

**start = (0, 0)**

**end = (4, 4)**

**path = astar(grid, start, end)**

**if path:**

**print("Path found:")**

**print(path)**

**else:**

**print("No path found.")**

**OUTPUT**

**A screenshot of a computer

AI-generated content may be incorrect.**

**CONCLUSION  
The robot traversal simulation demonstrates basic autonomous navigation in an obstacle-rich environment, highlighting the utility of algorithms like A\* in robotic path planning scenarios.**