Shri Vile Parle Kelavani Mandal's

INSTITUTE OF TECHNOLOGY

DHULE (M.S.)

DEPARMENT OF COMPUTER ENGINEERING

Subject : Artificial Intelligence Lab

Remark

Name : Mohammad Anas Aarif Baig Mirza **Roll No. :** 40

Class : B.tech Final Year **Batch : B2** **Division: -**

Signature

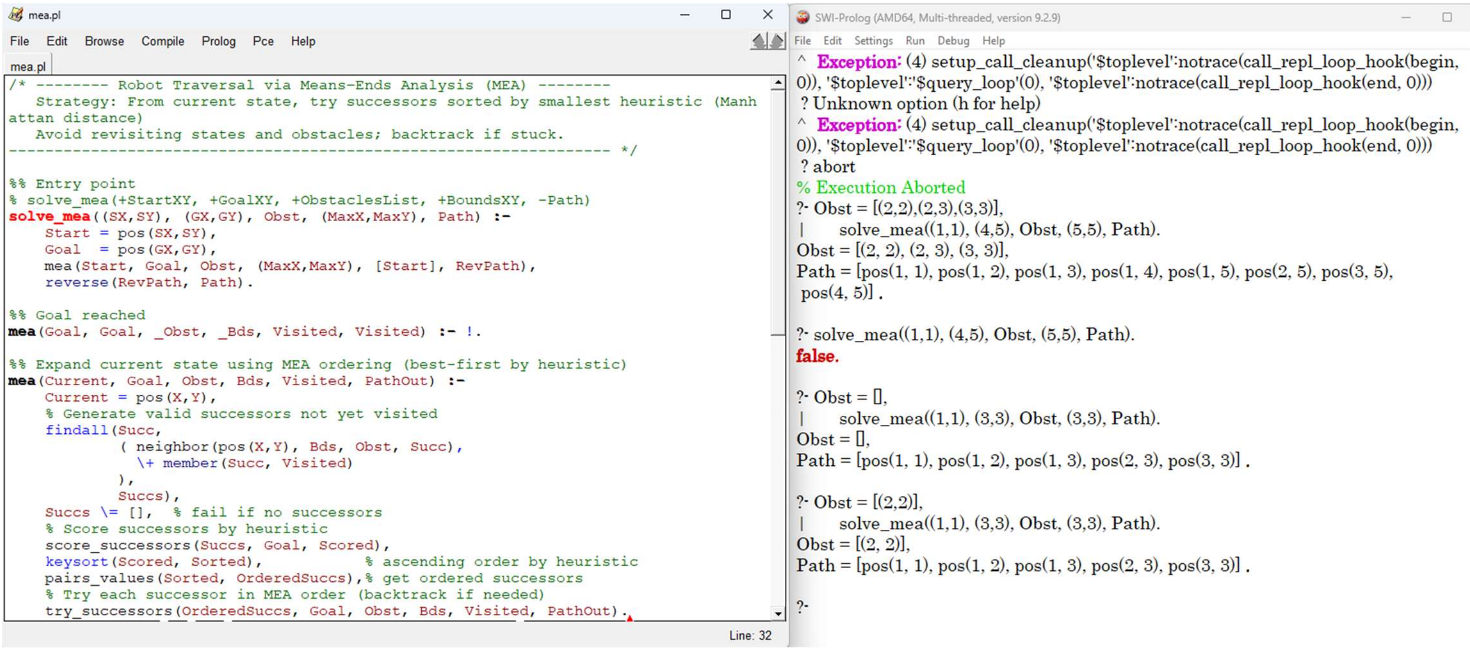
Expt. No. :06 **Date : 16/10/2025**

Title : Solve Robot (traversal) problem using means End Analysis.

Program Code:

```
/* -------- Robot Traversal via Means Ends Analysis (MEA) --------
   Strategy: From current state, try successors sorted by smallest heuristic (Manhattan distance)
   Avoid revisiting states and obstacles; backtrack if stuck.
---------------------------------------------------------------- */

%% Entry point
% solve_mea(+StartXY, +GoalXY, +ObstaclesList, +BoundsXY, -Path)
solve_mea((SX,SY), (GX,GY), Obst, (MaxX,MaxY), Path) :-
    Start = pos(SX,SY),
    Goal  = pos(GX,GY),
    mea(Start, Goal, Obst, (MaxX,MaxY), [Start], RevPath),
    reverse(RevPath, Path).

%% Goal reached
mea(Goal, Goal, _Obst, _Bds, Visited, Visited) :- !.

%% Expand current state using MEA ordering (best-first by heuristic)
mea(Current, Goal, Obst, Bds, Visited, PathOut) :-
    Current = pos(X,Y),
    % Generate valid successors not yet visited
    findall(Succ,
            ( neighbor(pos(X,Y), Bds, Obst, Succ),
              \+ member(Succ, Visited)
            ),
            Succs),
    Succs \= [],  % fail if no successors
    % Score successors by heuristic
    score_successors(Succs, Goal, Scored),
    keysort(Scored, Sorted),          % ascending order by heuristic
    pairs_values(Sorted, OrderedSuccs),% get ordered successors
    % Try each successor in MEA order (backtrack if needed)
    try_successors(OrderedSuccs, Goal, Obst, Bds, Visited, PathOut).

%% Try successors in order
try_successors([S|_], Goal, Obst, Bds, Vis, PathOut) :-
    mea(S, Goal, Obst, Bds, [S|Vis], PathOut).
try_successors([_|Ss], Goal, Obst, Bds, Vis, PathOut) :-
    try_successors(Ss, Goal, Obst, Bds, Vis, PathOut).
try_successors([], _Goal, _Obst, _Bds, _Vis, _PathOut) :- fail.

%% Generate 4-connected neighbors (N,S,E,W) within bounds and not obstacles
neighbor(pos(X,Y), (MaxX,MaxY), Obst, pos(X, Y1)) :-
    (Y1 is Y+1 ; Y1 is Y-1),
    within(1, MaxY, Y1),
    \+ blocked((X,Y1), Obst).
neighbor(pos(X,Y), (MaxX,MaxY), Obst, pos(X1, Y)) :-
    (X1 is X+1 ; X1 is X-1),
    within(1, MaxX, X1),
    \+ blocked((X1,Y), Obst).

%% Utility checks
within(Low, High, V) :- V >= Low, V =< High.
blocked((X,Y), Obst) :- member((X,Y), Obst).

%% Heuristic: Manhattan distance
h(pos(X,Y), pos(GX,GY), H) :-
    DX is abs(X-GX),
    DY is abs(Y-GY),
    H is DX + DY.

%% Score successors by heuristic
score_successors([], _Goal, []).
score_successors([S|Ss], Goal, [H-S|Rest]) :-
    h(S, Goal, H),
    score_successors(Ss, Goal, Rest).

%% Extract values from Key-Value pairs
pairs_values([], []).
pairs_values([_-V|T], [V|Vs]) :- pairs_values(T, Vs).
```

Output:



```
/* -------- Robot Traversal via Means-Ends Analysis (MEA) --------
   Strategy: From current state, try successors sorted by smallest heuristic (Manh
attan distance)
   Avoid revisiting states and obstacles; backtrack if stuck.
--------------------------------------------------------------- */

%% Entry point
% solve_mea(+StartXY, +GoalXY, +ObstaclesList, +BoundsXY, -Path)
solve_mea((SX,SY), (GX,GY), Obst, (MaxX,MaxY), Path) :-
    Start = pos(SX,SY),
    Goal  = pos(GX,GY),
    mea(Start, Goal, Obst, (MaxX,MaxY), [Start], RevPath),
    reverse(RevPath, Path).

%% Goal reached
mea(Goal, Goal, _Obst, _Bds, Visited, Visited) :- !.

%% Expand current state using MEA ordering (best-first by heuristic)
mea(Current, Goal, Obst, Bds, Visited, PathOut) :-
    Current = pos(X,Y),
    % Generate valid successors not yet visited
    findall(Succ,
            ( neighbor(pos(X,Y), Bds, Obst, Succ),
              \+ member(Succ, Visited)
            ),
            Succs),
    Succs \= [],  % fail if no successors
    % Score successors by heuristic
    score_successors(Succs, Goal, Scored),
    keysort(Scored, Sorted),            % ascending order by heuristic
    pairs_values(Sorted, OrderedSuccs),% get ordered successors
    % Try each successor in MEA order (backtrack if needed)
    try_successors(OrderedSuccs, Goal, Obst, Bds, Visited, PathOut).
```

Line: 32
SWI-Prolog (AMD64, Multi-threaded, version 9.2.9)
File Edit Settings Run Debug Help

```
^  Exception: (4) setup_call_cleanup('$toplevel':notrace(call_repl_loop_hook(begin,
0)), '$toplevel':'$query_loop'(0), '$toplevel':notrace(call_repl_loop_hook(end, 0)))
 ? Unknown option (h for help)
^  Exception: (4) setup_call_cleanup('$toplevel':notrace(call_repl_loop_hook(begin,
0)), '$toplevel':'$query_loop'(0), '$toplevel':notrace(call_repl_loop_hook(end, 0)))
 ? abort
% Execution Aborted
?- Obst = [(2,2),(2,3),(3,3)],
|     solve_mea((1,1), (4,5), Obst, (5,5), Path).
Obst = [(2, 2), (2, 3), (3, 3)],
Path = [pos(1, 1), pos(1, 2), pos(1, 3), pos(1, 4), pos(1, 5), pos(2, 5), pos(3, 5),
 pos(4, 5)] .

?- solve_mea((1,1), (4,5), Obst, (5,5), Path).
false.

?- Obst = [],
|     solve_mea((1,1), (3,3), Obst, (3,3), Path).
Obst = [],
Path = [pos(1, 1), pos(1, 2), pos(1, 3), pos(2, 3), pos(3, 3)] .

?- Obst = [(2,2)],
|     solve_mea((1,1), (3,3), Obst, (3,3), Path).
Obst = [(2, 2)],
Path = [pos(1, 1), pos(1, 2), pos(1, 3), pos(2, 3), pos(3, 3)] .

?-
```