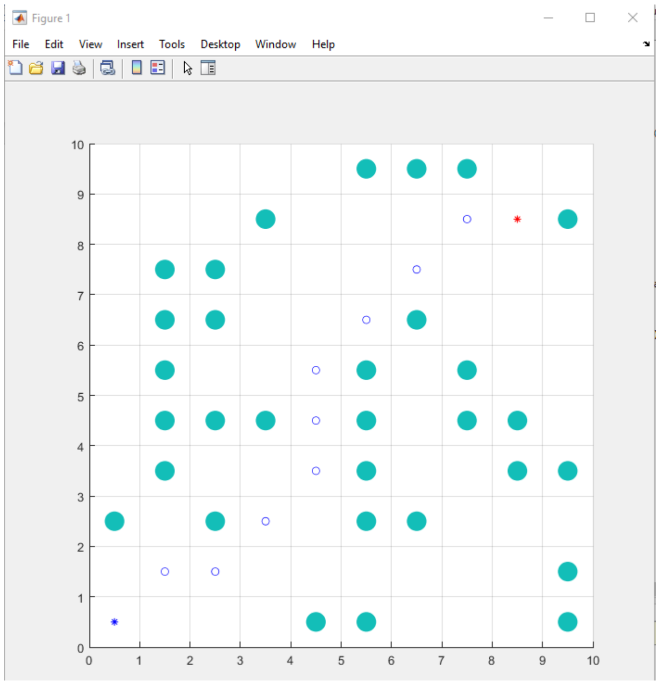
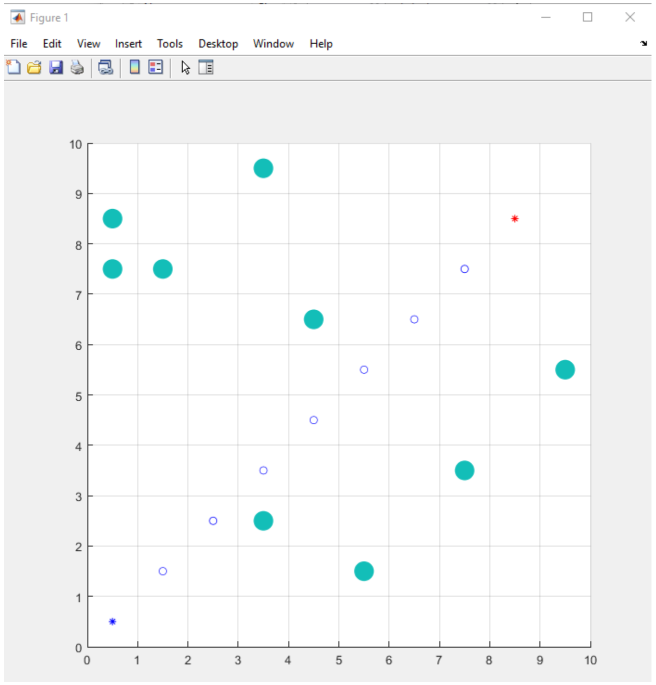
Chapter 2.1 Report for A* planning algorithms by MATLAB

User ID: rabbit5024

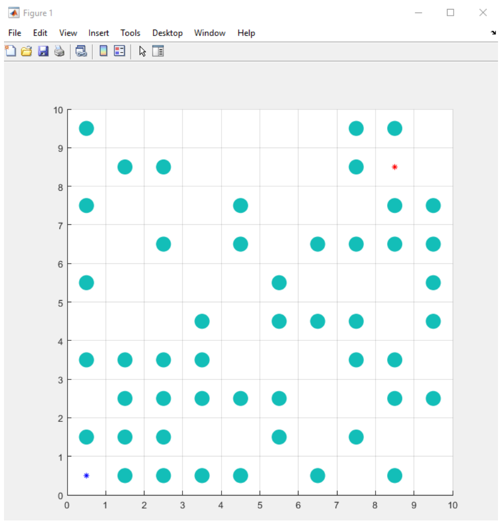
1. The screenshot of planning results for 3 random generated maps.
   a. MAP1 (obstacle_ratio = 0.25)



   b. MAP2(obstacle_ratio = 0.1)



   c. MAP3(obstacle_ratio = 0.5)

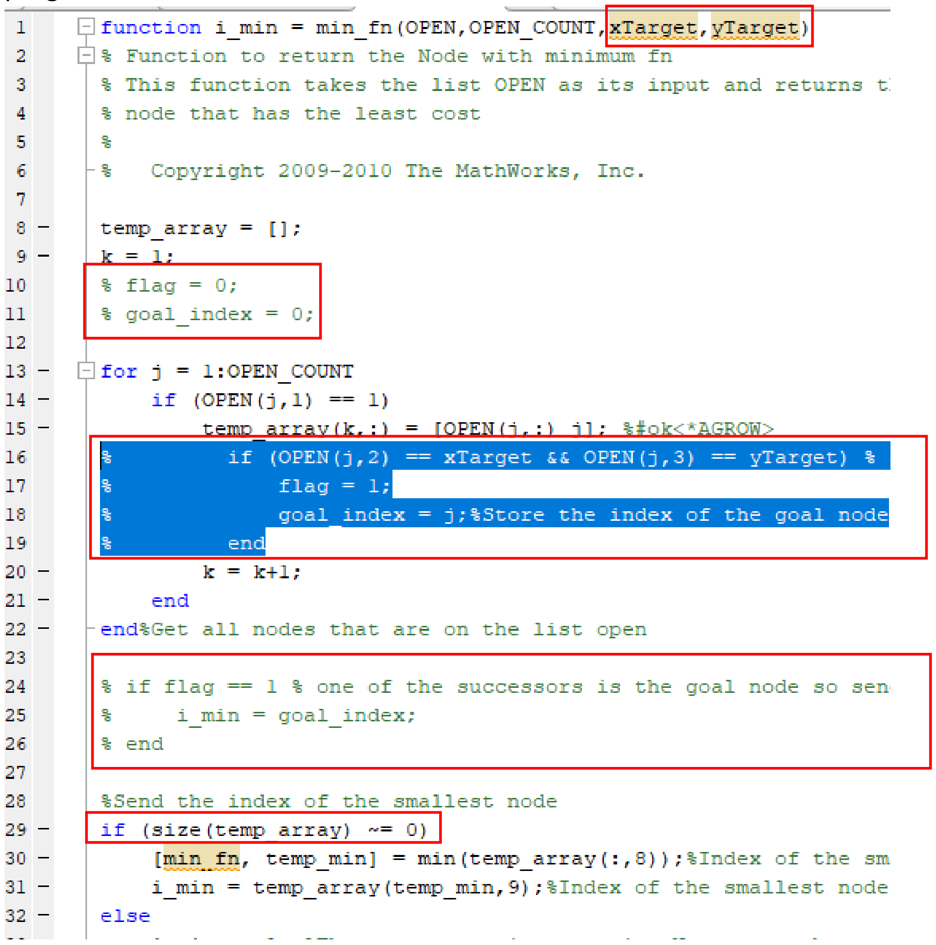


2. Results analysis

   At normal obstacle situation, the A* algorithm could find the optimal path with minimal distance cost. If there are too many obstacles and no path exist, the algorithm will return an empty path.

3. Others that are interests

   The 'min_fn' function requires the **xTarget** and **yTarget**, but I think it is unnecessary for the sort function to know the target point. I commented these variables in the subfunction, and the program still works well.



```
function i_min = min_fn(OPEN,OPEN_COUNT,xTarget,yTarget)
% Function to return the Node with minimum fn
% This function takes the list OPEN as its input and returns t
% node that has the least cost
%
%   Copyright 2009-2010 The MathWorks, Inc.

temp_array = [];
k = 1;
% flag = 0;
% goal_index = 0;

for j = 1:OPEN_COUNT
    if (OPEN(j,1) == 1)
        temp_array(k,:) = [OPEN(j,:) j]; %#ok<*AGROW>
%           if (OPEN(j,2) == xTarget && OPEN(j,3) == yTarget) %
%               flag = 1;
%               goal_index = j;%Store the index of the goal node
%           end
        k = k+1;
    end
end%Get all nodes that are on the list open

% if flag == 1 % one of the successors is the goal node so sen
%     i_min = goal_index;
% end

%Send the index of the smallest node
if (size(temp_array) ~= 0)
    [min_fn, temp_min] = min(temp_array(:,8));%Index of the sm
    i_min = temp_array(temp_min,9);%Index of the smallest node
else
```