%General inputs needed for the set-up

xy\_goal = [0.5,0.5];

xy\_start = [-0.5 -0.5];

n = 65; % number of random samples

r = 0.03; %robot radius = 3 cm -- to be used in collision detection

rand\_nodes = (1\* rand(n,2) )-0.5;

nodes = transpose([1:n+1]); % assigning IDs of nodes

edges = double.empty;

nodes(1,2:3) = [xy\_start]; %adding starting node to nodes matrix

nodes (2:end,2:3) = rand\_nodes; %adding the random nodes to nodes matrix

nodes (end:end,2:3) = xy\_goal;

path = int16.empty;

Tree(1,:) = [1 xy\_start];

obstacles = csvread('obstacles.csv',5,0);

%converting 10% of the random sample nodes to be the goal node

rand\_goal = randi([2 n],1,ceil(n\*0.1));

for i=1:(n\*0.1)

row\_no = rand\_goal(:,i);

nodes(row\_no,2:3) = xy\_goal;

end

%start of the RRt algorithm

while length(Tree) < n

%picking a random sample point

sample\_point = nodes(randi([2 n],1),:);

closest\_point = dsearchn(Tree(:,2:3),sample\_point(1,2:3)); %returning the row of nearest point in the tree

closest\_point = Tree(closest\_point,:);

%checking collision

[boolean\_collision, sample\_point] = collision\_check (sample\_point,closest\_point,obstacles,r);

if boolean\_collision == 0

Tree = [Tree; sample\_point]; %adding the sample node to the tree

edge\_initial = sqrt((sample\_point(1,2)-closest\_point(1,2))^2+(sample\_point(1,3)-closest\_point(1,3))^2);

edges = [edges; closest\_point(1,1) sample\_point(1,1) edge\_initial];%creating the edges matrix

parent\_node(sample\_point(1,1),1) = closest\_point(1,1);

%finding the path

if sample\_point(1,2:3) == xy\_goal

last\_node = sample\_point(1,1);

path = last\_node;

while last\_node ~= nodes(1,1)

path = [parent\_node(last\_node,1) path];

last\_node = parent\_node(last\_node,1);

end

csvwrite('edges.csv',edges,0,0);

csvwrite('nodes.csv',nodes,0,0);

csvwrite('Path.csv',path,0,0);

path

display('Success!')

break;

end

end

end