Appendix A

Matlab codes

Function occ.map.m

```
function  occ_map(omap3D,start,goal)

q_start = start;
q_final = goal;


numofobstacles=6;

global mapWidth
global mapLength
global mapHeight

mapWidth = 200;
mapLength = 200;
mapHeight = 200;


width       =   [ 25  30   5  60  50  20];
length      =   [ 25   6  40  12  50  20];
height      =   [ 90 180 145 130 200  70];
xPosition   =   [ 25  70 180 100  25 150];
yPosition   =   [ 50  25 145 111 130  50];

for i=1:1:numofobstacles
[xObstacle,yObstacle,zObstacle] = meshgrid(xPosition(i):xPosition(i)+
width(i),yPosition(i):yPosition(i)+length(i),0:height(i));

xyzObstacles = [xObstacle(:) yObstacle(:) zObstacle(:)];
```

```
setOccupancy(omap3D,xyzObstacles,1)


end
[xGround,yGround,zGround] = meshgrid(0:mapWidth,0:mapLength,0);
xyzGround = [xGround(:) yGround(:) zGround(:)];
setOccupancy(omap3D,xyzGround,1)

figure("Name","3D Occupancy Map")
fig.Color = [0 0.8 0.8];

show(omap3D)
hold on
plot3(q_start(1), q_start(2), q_start(3),'bo','MarkerFaceColor','yellow',
'MarkerSize',8)
hold on
plot3(q_final(1), q_final(2), q_final(3),'bo','MarkerFaceColor','yellow',
'MarkerSize',8)


end
```

Function q.rand.gen.m

```
function [points_rdmap] = q_rand_gen(iter,q_start, q_final,omap3D)


global mapWidth
global mapLength
global mapHeight

points_rdmap(1,:) = [q_start(1) q_start(2) q_start(3)];

 s=2;

 for i=1:1:iter

qrand = random(mapWidth, mapLength, mapHeight);
```

```
15
    if( checkOccupancy(omap3D, qrand) == -1)
17      points_rdmap(s,:) = [qrand(1) qrand(2) qrand(3)];
        s = s+1;
19   end

21  end

23  points_rdmap(s,:) = [q_final(1) q_final(2) q_final(3)];
clear length
25


27 end
```

Function seg.gen.m

```
1 function [tree] = seg_gen(ds, points_rdmap,omap3D)
ret =1;
3
indx = [];
5 indx(1,:) = [-1 -1];

7 for j = 1:1:length(points_rdmap)

9     cnd = points_rdmap(j,:);

11    for m  = 1:1:length(points_rdmap)

13        flag = 0;

15        cnd_s = points_rdmap(m,:);

17        ds_i = distance_3d(cnd,cnd_s);

19            for  v =1:1:length(indx(:,1))

21                if(indx(v,2) == j && indx(v,1) == m)
                          flag = 1;
23                end
```

```
25                end
                 if(ds_i > 0 && ds_i < ds && flag ==0 )
27
                     indx(ret,:)=[j m];
29                   ret = ret+1;

31               end


33
   end
35 end

37 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

39
   l =1 ;
41 for p = 1:1:length(indx)

43      first = points_rdmap(indx(p,1),:);
       second = points_rdmap(indx(p,2),:);
45     ok = 1;

47 pts = segm(first,second);

49 for k = 1:1:length(pts)
   point = pts(k,:);
51
       if(checkOccupancy(omap3D, point) == 1)
53         ok = 0;
           break
55
       end
57


59
   end
61
   if (ok ==1)
63     tree(l,:) = [ indx(p,1) indx(p,2)];
       l=l+1;
65
   end
67
   end
```

```
69
end
```

Function segm.m

```
function pts = segm(first, second)
2
t = 100;
4
seg = first(1) : (second(1) - first(1))/t : second(1);
6 if(length(seg) == 0)
    seg = first(1)*ones(1,t+1);
8 end

10 seg1 = first(2) : (second(2) - first(2))/t : second(2);
if(length(seg1) == 0)
12     seg1 = first(2)*ones(1,t+1);
end
14
seg2 = first(3) : (second(3) - first(3))/t : second(3);
16 if(length(seg2) == 0)
    seg2 = first(3)*ones(1,t+1);
18 end


20
for k = 1:1:length(seg)
22 pts(k,:) = [seg(k)  seg1(k)  seg2(k)];
end
24
end
```

Function A.star.m

```
function [ tree_dfs , sol_found] = A_star(points_rdmap,tree)


%%%%%%%%%%ADJ MATRIX%%%%%%%%%%%%%%%%%
ADJ = zeros(length(points_rdmap), length(points_rdmap) );

for p = 1:1:length(points_rdmap)

    for s=1:1:length(tree)
        if(tree(s,1) == p)
            ADJ(p,tree(s,2)) =distance_3d(points_rdmap(p,:) ,
            points_rdmap(tree(s,2),:));

        end

    end


end
goal = length(points_rdmap);
sol_found = 0;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

 for i =1:1:length(ADJ(1,:))
 nodes(i,:) = [i 0];
 end

OPEN  =[];
final = [];
OPEN(1,:) = [1 0 0];
nodes(1,2) = 1;

N_best = OPEN(1,:);
final(end+1,:)  = [1 0 0];
OPEN(1,:) = [];

for p =1:1:length(ADJ(1,:))
    if(ADJ(N_best(1),p) > 0)
        OPEN(end+1,:)  =  [p N_best(1)  ADJ(N_best(1),p) ];
        final(end+1,:) =  [p N_best(1)  ADJ(N_best(1),p)];
        nodes(p,2) =1;
    end
end
```

```
45
   OPEN = sortrows(OPEN,3);
47
   %%%%%%%%%%%%%%%%%%%%%%
49

51 while(1)

53 if(length(OPEN(:,1)) == 0)
       break
55 end

57 N_best = OPEN(1,:);

59
   if (N_best(1) == goal)
61     sol_found =1;
       break
63 end

65 OPEN(1,:) = [];

67 for p =1:1:length(ADJ(1,:))
       if(ADJ(N_best(1),p) > 0)
69
           if(nodes(p,2) == 0 )
71
           OPEN(end+1,:) =  [p N_best(1)  (ADJ(N_best(1),p) + N_best(3)) ];
73         final(end+1,:) = [p N_best(1)  (ADJ(N_best(1),p) + N_best(3))];
           nodes(p,2) =1;
75
           else
77
               for n=1:1:length(final)
79                 if(final(n) == p)
                       cost_p = final(n,3);
81                 end
               end
83
            flag =  N_best(3) + ADJ(N_best(1), p);
85          if(flag < cost_p )
                 for n=1:1:length(final)
87                 if(final(n) == p)
                       final(n,2) = N_best(1);
89                 end
```

```
            end
91
        end
93
    end
95 end
end
97 OPEN = sortrows(OPEN,3);
end
99

101 if(sol_found == 1)

103 ind = goal;
tree_dfs(1) = goal;
105


107

109 while (ind > 1)

111 for n=1:1:length(final)

113     if(final(n,1) == ind)

115         ind = final(n,2);
            tree_dfs(end+1) = final(n,2);
117
     end
119         end
                end
121                 end

123 tree_dfs = flip(tree_dfs);

125
end
```

Function draw.m

```
function draw(points_rdmap,vec,tree,sol_found)

if (sol_found == 1)

% for p =1:1:length(tree(:,1))
%
% f_c_1 = points_rdmap(tree(p,1),1);
% s_c_1 = points_rdmap (tree(p,1),2);
% t_c_1 = points_rdmap (tree(p,1),3);
%
% f_c_2 = points_rdmap(tree(p,2),1);
% s_c_2 = points_rdmap (tree(p,2),2);
% t_c_2 = points_rdmap (tree(p,2),3);
%
% hold on
%  plot3( f_c_1,s_c_1, t_c_1, 'x', 'Color',  [0 0.4470 0.7410])
%  hold on
%  plot3(f_c_2,s_c_2, t_c_2, 'x', 'Color',  [0 0.4470 0.7410])
%  hold on
%  line([f_c_1 , f_c_2], [s_c_1 , s_c_2], [t_c_1 , t_c_2], 'Color', 'g')
%  drawnow
% end

for p=1:1: length(vec)-1

point_one = points_rdmap(vec(p),:);
point_two = points_rdmap(vec(p+1),:);

hold on
line([point_one(1), point_two(1)], [point_one(2), point_two(2)], [point_one(3),
point_two(3)], 'Color', 'r', 'LineWidth', 3)
drawnow

end

else
    display('Solution␣not␣found!')
    end
        end
```

Function distance.3d.m

```
function d = distance_3d(q1,q2)
    d = sqrt((q1(1)-q2(1))^2 + (q1(2)-q2(2))^2 + (q1(3)-q2(3))^2);
end
```

Function random.m

```
function qrand = random(x_map,y_map,z_map)

    qrand=[floor(rand*x_map), floor(rand*y_map) , floor(rand*z_map)];

end
```

Function multi.waypts.m

```
clear all
close all
clc

%%%%Waypoints%%%%%%%%%%%%%%%

path(1,:)  =  [  1    1    1];
path(2,:)  =  [ 70   60   30];
path(3,:)  =  [ 40   90   50];
path(4,:)  =  [ 20   60   50];
path(5,:)  =  [ 40   40   50];
path(6,:)  =  [ 140  40   40];
path(7,:)  =  [180   60   40];
path(8,:)  =  [100  100  100];
path(9,:)  =  [ 90  160  100];
path(10,:)  =  [ 40  190  100];
path(11,:) =  [ 10  160  100];
```

```
path(12,:) =  [ 40  120  150];
path(13,:) =  [140  150  120];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

sol_found = 0;
q_start = path(1,:);
q_final = path(13,:);

omap3D = occupancyMap3D;
occ_map(omap3D,q_start, q_final);

for p =2:1:length(path)-1
    pt = path(p,:);
 plot3(pt(1), pt(2), pt(3),'bo','MarkerFaceColor','yellow','MarkerSize',8)
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

iter  = 300;
ds    = 100 ;
vec_f = [q_start];

points_rdmap = q_rand_gen(iter,q_start,  q_final,omap3D);

for k =1:1:length(path)-1

sol_found = 0;

points_rdmap(1,:)   = path(k,:);
points_rdmap(end,:) = path(k+1,:);

tree = seg_gen(ds,points_rdmap,omap3D);

[vec ,  sol_found] = A_star(points_rdmap,tree);

if(sol_found ==1)

for  i =1:1:length(vec)
tmp(i,:) = points_rdmap(vec(i),:);
end

if(vec_f(end,:) == tmp(1,:))
    tmp(1,:) = [];
end
```

```
63  vec_f = [vec_f;tmp];

65  end


67
    draw(points_rdmap,vec,tree,sol_found);
69
    if(sol_found == 0)
71      close all
        clear final_points
73      break
    end
75
    end
```