APSC 200 P2 Project script

Table of Contents

Section 203 Group 03 Fall 2016

Initial Declarations

```
%Clear variables and close plots
close all;
clear all;

%Importing population distribution data
mm = importdata('PopDistDataDoc.mat');
sec = []; %Initializing array for number of people per sector

%Declaring number of Robots and itterations
nbots = 30; %Number of robots
nitts = 50; %Number of itterations

%Generating random start positions
for i = 1:nbots
    Px(i,1) = length(mm(1,:)) + 25*rand() - 25; %X position
    Py(i,1) = length(mm(:,1))/2 + 25*rand() - 13; %Y Position
end

%Converting Initial x and y to combined vector
X(:, 1) = Px;
X(:, 2) = Py;

%Generating grid values
xpt = 1:length(mm(1,:));
ypt = 1:length(mm(:,1));

%Declaring bounds on region
bnd =
 [0,0;0,length(mm(:,1));length(mm(1,:)),length(mm(:,1));length(mm(1,:)),0];
```

Main program loop

```
for itt = 1:nitts

    %Calculating bound Voronoi diagram
```

```
    [V,R]=VoronoiBounded(Px,Py, bnd); %NOTE: This is a downloaded
function
```

Calculating center of mass for each region

```
    for i = 1:nbots
        figure(itt) %Creates a new figure for each itteration
        hold on %Allow multiple plots on each figure

        %Plotting and storing bound region
        P = patch(V(R{i},1),V(R{i},2),i);
        plot(Px,Py,'.r') %Plotting drone location

        %Defining x and y region for optimised point check
        XRegion = V(R{i}, 1);
        YRegion = V(R{i}, 2);

        %Finding minimum and maximum values in given region
        minx = round(min(XRegion));
        miny = round(min(YRegion));
        maxx = round(max(XRegion));
        maxy = round(max(YRegion));

        %Check to ensure no zero division
        if(minx == 0)
            minx = 1;
        end
        if(miny == 0)
            miny = 1;
        end

        %Declaration of all-zero matrix same dimmensions as complete
area
        m = zeros([length(mm(:,1)) length(mm(1,:))]);

        %Collection of points within bound region
        for j = minx:maxx
            for k = miny:maxy
                %If a given point is within the region it is added to
the
                %matrix to be calculated
                if(inpolygon(xpt(j), ypt(k), P.XData, P.YData))
                    m(k, j) = mm(k, j);
                end
            end
        end

        %Declaring variables
        smx = 0;
        smy = 0;

        %Calculating x CM
        for j = 1:length(xpt)
```

```
            dsmx(j) = sum(m(:,j))*xpt(j); %Weighted sum of columns
            smx = smx + sum(m(:,j)); %Sum of columns
        end
        if(smx == 0) %Check to ensure no zero division
            smx = 1;
        end
        cmx = sum(dsmx)/smx; %Horizontal center of mass calculation

        %Calculating y CM
        for j = 1:length(ypt)
            dsmy(j) = sum(m(j,:))*ypt(j); %Weighted sum of rows
            smy = smy + sum(m(j,:)); %Sum of rows
        end
        if(smy == 0) %Check to ensure no zero division
            smy = 1;
        end
        cmy = sum(dsmy)/smy; %Vertical center of mass calculation

        %Storing number of people in each region on last itteration
        if(itt == nitts)
            sec(i) = smx;
        end

        %New coordinate for drone on next itteration (current CM)
        nx(i, 1) = cmx;
        ny(i, 1) = cmy;

        %Plot CM for region (Star)
        plot(cmx,  cmy, 'rp')

        %Clear arrays to prevent errors
        clear dsmy
        clear dsmx
        clear smx
        clear smy
    end
```

Calculating coverage on each itteration

```
    %Declaring all zero array same size as total area
    cov = zeros(length(m(:,1)), length(m(1,:)));
    area = zeros(length(m(:,1)), length(m(1,:)));

    for i = 1:nbots
        %Generating and plotting 'coverage' region around each drone
        [bits] = circ([Px(i) Py(i)], 20); %Generating points
        plot(bits(:,1), bits(:,2), 'k'); %Plotting circle

        %Checking number of points in each region
        for j = 1:length(xpt)
            for k = 1:length(ypt)
                if(inpolygon(xpt(j), ypt(k), bits(:,1), bits(:,2)) &&
 mm(k, j) > 1)
```

```
                    cov(k, j) = 1;
                    area(k, j) = 1;
                elseif(mm(k, j) > 1)
                    area(k, j) = 1;
                end
            end
        end
    end

    %Coverage calculation
    coverage(itt) = sum(sum(cov))/sum(sum(area)) * 100;
```

Plot formatting

```
    axis([0 length(mm(1,:)) 0 length(mm(:,1))])
    title(['Iteration ' num2str(itt)]);
    xlabel('Longitude');
    ylabel('Latitude');

    hold off;
    Px = nx;
    Py = ny;

end
```

Plotting of other stats

```
%Plot of number of people per regino
figure
scatter(1:length(sec), sec.* 0.1, '.')
title('Number of people in each sector');
xlabel('Drone');
ylabel('Number of people in region');

%Plot of population coverage on each itteration
figure
scatter(1:length(coverage), coverage,  '.')
title('Coverage of populated areas');
xlabel('Number of itterations');
ylabel('Coverage (%)');

%Mesh plot of population distribution
figure
mesh(mm)
axis([0 226 0 226 0 3*10^4])
xlabel('Longitude');
ylabel('Lattitude');
zlabel('Number of people (per km^2)');
```

Published with MATLAB® R2016b