Hacettepe University - Computer Engineering

Swarm Systems CMP 756

Homework-2

Student Name	Ayça KULA
Student ID:	202237285

Spring 2022

1 Assign random initial states to every agent in a bounded region

Random initial states which are polar coordinates are assigned to every agent in the system using the "rand" command. Then, these states are converted into cartesian coordinates using the "pol2cart" command in MATLAB. The design of initial states can be seen in algorithm 1.

```
%% Assign random initial states to every agent in a bounded region
% Define location in polar coordinates for the agents
R_rand = rand(1, num_agent) * 20;
theta = 2 * pi* rand(1, num_agent);  % Must be in radians

%% For the steady-state, define the radius of the hyperbola
% for the convergent area,
% Define a starting radius
R_th = 5;
R_start = 70;

%% Number of states (x & y are our states)
numOfStates = 2;
state_vec = zeros(num_agent,numOfStates,iter_num);
state_x = zeros(num_agent,iter_num);
for i = 1:num_agent
    % Define location in Cartesian coordinates
    [x(i), y(i)] = pol2cart(theta(i), R_start + R_rand(i));
    % Initial State vector for each agent
    state_vec(i,:,1) = [x(i), y(i)];
end
```

Algorithm 1: Assignment of random initial states

2 Design J(x), g(x) for aggregation problem

The potential function is given as:

$$J(x) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \left[\frac{a}{2} \left\| x_i - x_j \right\|^2 + \frac{bc}{2} \exp\left(-\frac{\left\| x_i - x_j \right\|^2}{c} \right) \right] \tag{1}$$

And its its corresponding attraction/repulsion function can be calculated as:

$$g(y) = -y \left[a - b \exp\left(-\frac{\|y\|^2}{c} \right) \right] \tag{2}$$

The implementation of corresponding attraction/repulsion function can be seen in 2. Firstly, the a,b and c constants are defined with respect to the simulation done in [1].

```
%% Define positive finite constants
% linear attraction constant,repulsion bound
a = 1; b = 20; c = .2;
% Eqiuilibrium distance
delta = sqrt(c*log(b/a));

%% Initialize control input
controlInput = zeros(num_agent,numOfStates,iter_num);
attrRepulFunc = zeros(num_agent,numOfStates,iter_num);
centroid = zeros(numOfStates,iter_num);

%% For each iteration
% i,j indicates different states
for iter = 1:iter_num
    for i = 1:num_agent
        %if (norm(state_vec(i,:,iter)) > R_th) % Convergent radius
            for j = 1:num_agent
                if(i~=j)
                    % Difference of agents x values in thestate vector
                    stateDiff(i,:,iter) = state_vec(i,:,iter)-state_vec(j,:,iter);

                    % g(x)-->(attraction/repulsion function) aggregation problem
                    attrRepulFunc(i,:,iter) = (a - b*exp(-sum(stateDiff(i,:,iter).^2) / c));

                    % Control input
                    controlInput(i,:,iter) = controlInput(i,:,iter) - stateDiff(i,:,iter).*attrRepulFunc(i,:,iter);

                    % The state difference is equal to control input u_i
                    state_vec(i,:,iter+1) = state_vec(i,:,iter) + controlInput(i,:,iter)*ts;
                end
            end
        %end
        % In order to find the centroid calculate the summed states
        centroid(:,iter) = centroid(:,iter) + state_vec(i,:,iter)';
    end
    % Find the centroid of agents
    centroid(:,iter) = centroid(:,iter)/num_agent;
end
```

Algorithm 2: Design J(x), g(x)

3 For the steady-state, define the radius of the hyperbola for the convergent area

Theorem II states that:
As time progresses all the members of the swarm will converge to a hyperball

$$B_\varepsilon(\bar{x}) = \{x : \|x - \bar{x}\| \leq \varepsilon\} \tag{3}$$

where

$$\varepsilon = \frac{b}{a}$$

Moreover, the convergence will occur in finite time bounded by

$$\bar{t} = \max_{i \in \{1,\ldots,N\}} \left\{ -\frac{1}{2a} \ln \left(\frac{\varepsilon^2}{2V_i(0)} \right) \right\}.$$

Since we had defined our a,b and c constants in previous section.We know that the hyperball will converge to $\varepsilon = \frac{b}{a} = \frac{20}{1} = 20$.As seen in figure 1, the distance from centroid is decreasing and finally it reaches under ε value.

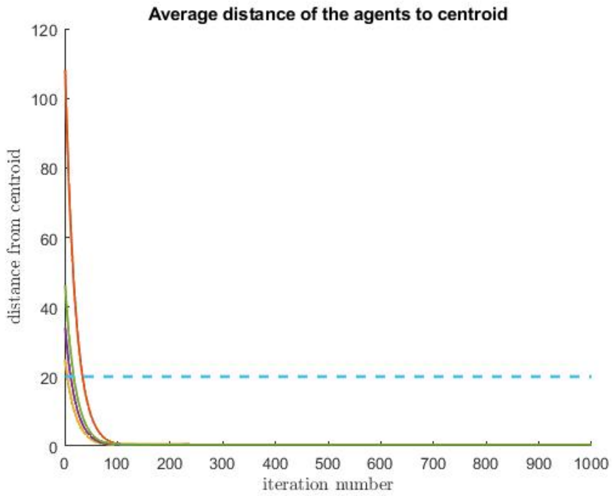


Figure 1: Convergence to hyperball

4 Validate Theorem I-II and Lemma I of the reference book

According to [1]:
Theorem I:
Consider a swarm consisting of agents with dynamics in (3.1) and with control input in (3.4) with an attraction/repulsion function g($\cdot$) which is odd and satisfies Assumption 1. For any $x(0) \in \mathbb{R}^{Nn}$, $t \to \infty$ we have $x(t) \to \Omega_e$.
Theorem-1 is satisfied as can be seen in figure 2.

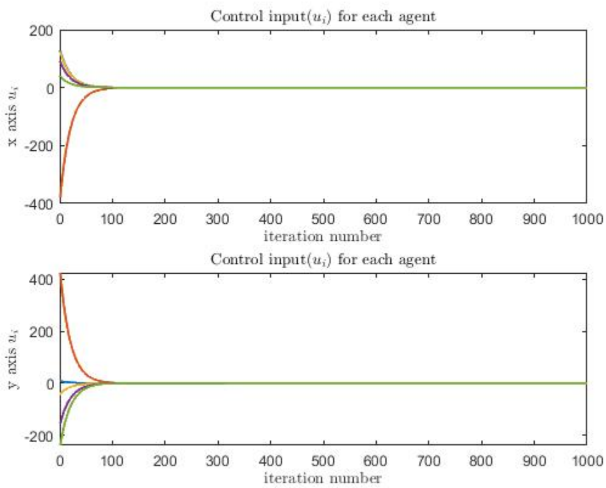


Figure 2: Control input converge to zero for the five agents

Lemma I:
The centroid x of the swarm consisting of agents with dynamics in (3.1) and with control input in (3.4) with an attraction/repulsion function g($\cdot$) which is odd and satisfies Assumption 1 is stationary for all t.

Lemma-1 is satisfied as can be seen in figure 3.

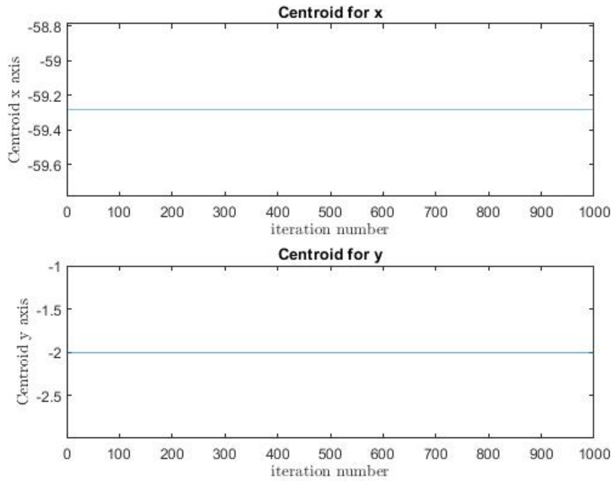


Figure 3: Centroids for both x and y axis

5 Plot initial positions and final positions of every agent by tracking the path on a single graph

The paths for the different five agents can be seen from figure 4. You can see that they converge to the centroid as in figure 5.

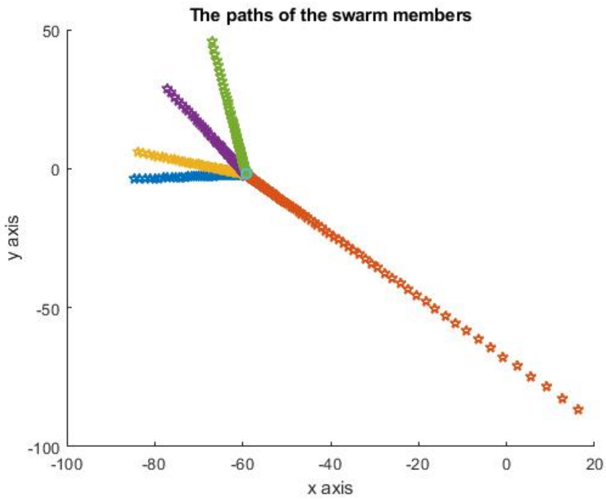


Figure 4: Paths/States(x,y) for the five agents

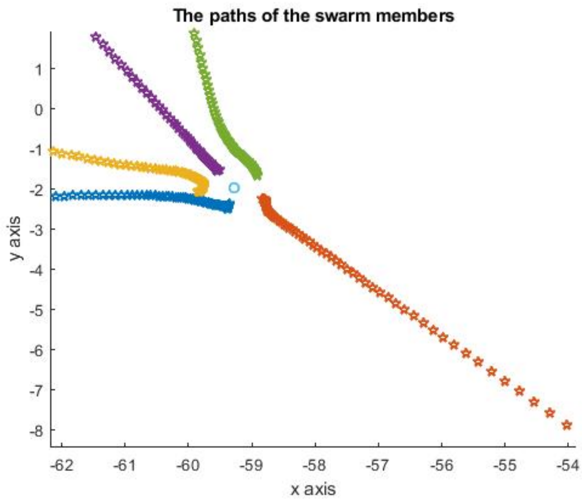


Figure 5: Converge to centroid

6 Discuss the results

It is assumed that the agents have a safety area of radius $\eta = 1$ and therefore keep a distance of at least $2\eta = 2$ units apart from each other.As can be seen from the figure 5, the agents do not collide with each other.As one can see from the figure, the minimum distance between agents is greater than $2\eta = 2$. Finally, it can be said that as the number of individuals increases, while the minimum distance between agents continues to be greater than $2\eta = 2$, the maximum distance scales with the number of agents. Therefore, we say that the swarm size scales with the number of agents while the density of the swarm remains relatively constant.
Also, we can manipulate the a,b and c constants in the code.From, formula given as:

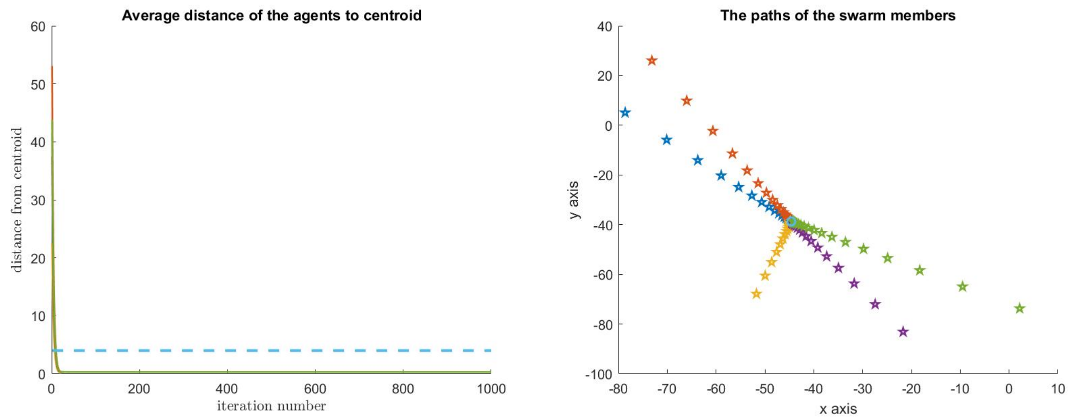
$$g_a\left(\|x_i - x_j\|\right) \geq \frac{a}{\|x_i - x_j\|} \tag{4}$$

we expect that if we **increase a**, the attraction function has to increase.Also, from figures 6a and 6b we see that the system converges more faster to stationary points.

On the other hand, if we decrease **increase b** from the formula below we see that the repulsion function decreases.

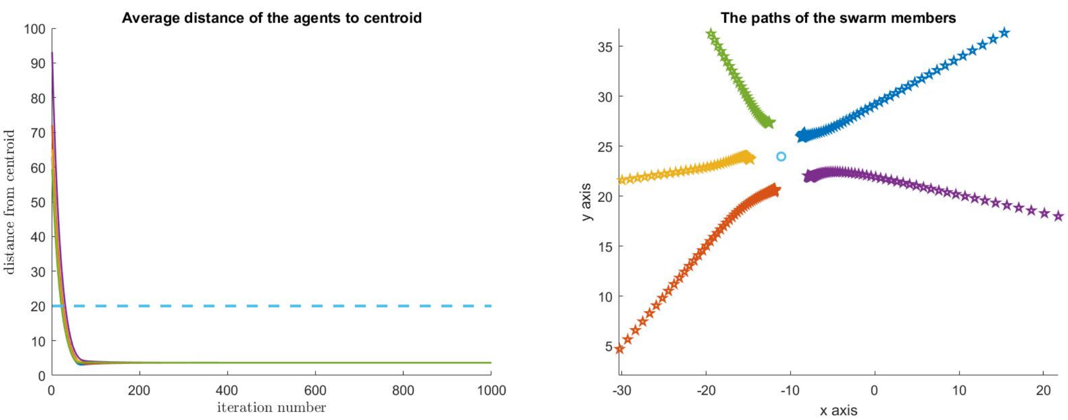
$$g_r\left(\|x_i - x_j\|\right) \leq \frac{b}{\|x_i - x_j\|^2} \tag{5}$$

If we **increase c**, then the distance between centroid and agents increase as seen in 7a and 7b. Further studies can be done using the code in the 6.



(a) Distance to centroid (b) States for the agents

Figure 6: The value of a is increased to 5



(a) Distance to centroid (b) States for the agents

Figure 7: The value of c is increased to 10

Appendix

Simulation code

```
%% SWRAM SYSTEM - HOMEWORK#2
clear all; close all;
bdclose all;
%% Define parameters for the system
num_agent = 5; % number of agents
iter_num = 1000; % number of iteration
ts = 0.01;
% time interval
% time = [0:ts:(ts*iter_num)];

%% Assign random initial states to every agent in a bounded region
% Define location in polar coordinates for the agents
R_rand = rand(1, num_agent) * 20;
theta = 2 * pi* rand(1, num_agent);  % Must be in radians

%% For the steady-state, define the radius of the hyperbola for the
    convergent area,
% Define a starting radius
R_th = 5;
R_start = 70;

%% Number of states (x & y are our states)
numOfStates = 2;
state_vec = zeros(num_agent,numOfStates,iter_num);
state_x = zeros(num_agent,iter_num);
for i = 1:num_agent
    % Define location in Cartesian coordinates
    [x(i), y(i)] = pol2cart(theta(i), R_start + R_rand(i));
    % Initial State vector for each agent
    state_vec(i,:,1) = [x(i), y(i)];
end

%% Define positive finite constants
% linear attraction constant,repulsion bound
a = 1; b = 20; c = .2;
% Eqiuilibrium distance
delta = sqrt(c*log(b/a));

%% Initialize control input
controlInput = zeros(num_agent,numOfStates,iter_num);
attrRepulFunc = zeros(num_agent,numOfStates,iter_num);
centroid = zeros(numOfStates,iter_num);

%% For each iteration
% i,j indicates different states
for iter = 1:iter_num
    for i = 1:num_agent
        %if (norm(state_vec(i,:,iter)) > R_th) % Convergent radius
            for j = 1:num_agent
                if(i~=j)
                    % Difference of agents x values in thestate
```

```
    vector
                    stateDiff(i,:,iter) = state_vec(i,:,iter)-
    state_vec(j,:,iter);

                    % g(x)-->(attraction/repulsion function)
    aggregation problem
                    attrRepulFunc(i,:,iter) = (a - b*exp(-sum(
    stateDiff(i,:,iter).^2) / c));

                    % Control input
                    controlInput(i,:,iter) = controlInput(i,:,iter)
     - stateDiff(i,:,iter).*attrRepulFunc(i,:,iter);

                    % The state difference is equal to control
    input u_i
                    state_vec(i,:,iter+1) = state_vec(i,:,iter) +
    controlInput(i,:,iter)*ts;
                end
            end
        %end
        % In order to find the centroid calculate the summed states
        centroid(:,iter) = centroid(:,iter) + state_vec(i,:,iter)';
    end
    % Find the centroid of agents
    centroid(:,iter) = centroid(:,iter)/num_agent;
end

%% Validate Theorem I-II and Lemma I
% Lemma #1 --> Define centroid of the swarm
iteration = 1:iter_num;
figure
subplot(2,1,1)
% Plot centroid for x axis
plot(iteration,centroid(1,:))
ylim([centroid(1,1)-0.5 centroid(1,1)+0.5])
title('Centroid for x')
xlabel('iteration number','Interpreter','latex','Linewidth',2);
ylabel('Centroid x axis','Interpreter','latex','Linewidth',2);
hold on

% Plot centroid for y axis
subplot(2,1,2)
plot(iteration,centroid(2,:))
title('Centroid for y')
ylim([centroid(2,1)-1 centroid(2,1)+1])
xlabel('iteration number','Interpreter','latex','Linewidth',2);
ylabel('Centroid y axis','Interpreter','latex','Linewidth',2);


%% Theorem I --> Plot control input

figure;
hold all;
for k = 1:num_agent
    subplot(2,1,1)
    controlInput_x = squeeze(controlInput(k,1,:));
```

```
    plot(iteration,controlInput_x,'Linewidth',1.25);
    hold on;
    subplot(2,1,2)
    controlInput_y = squeeze(controlInput(k,2,:));
    plot(iteration,controlInput_y,'Linewidth',1.25);
    hold on;
end
subplot(2,1,1)
xlabel('iteration number','Interpreter','latex','Linewidth',1.25);
ylabel('x axis $u_{i}$','Interpreter','latex','Linewidth',1.25);
title('Control input($u_{i}$) for each agent','Interpreter','latex'
    ,'Linewidth',1.25);

subplot(2,1,2)
xlabel('iteration number','Interpreter','latex','Linewidth',1.25);
ylabel('y axis $u_{i}$','Interpreter','latex','Linewidth',1.25);
title('Control input($u_{i}$) for each agent','Interpreter','latex'
    ,'Linewidth',1.25);

%% Theorem II --> Swarm will converge to a hyperball
% The average distance to the center
figure;
hold all;
for k = 1:num_agent
    x_vec = squeeze(state_vec(k,1,2:end));
    x_c = centroid(1,:)';
    y_vec = squeeze(state_vec(k,2,2:end));
    y_c = centroid(2,:)';
    dis = sqrt((x_vec-x_c).^2+(y_vec-y_c).^2);
    plot(iteration, dis,'Linewidth',1.3);
    hold on
end
plot([iteration(1),iteration(end)],[b/a, b/a],'--','Linewidth',2);

xlabel('iteration number','Interpreter','latex','Linewidth',2);
ylabel('distance from centroid','Interpreter','latex','Linewidth'
    ,2);
title('Average distance of the agents to centroid','Linewidth',5);

%% Plot initial positions and final positions of every agent by
    tracking the path on a single graph
figure;
hold all;
for k = 1:num_agent
    x_vec = squeeze(state_vec(k,1,:));
    y_vec = squeeze(state_vec(k,2,:));
    plot(x_vec,y_vec,'p','Linewidth',1.25);
    hold on
end
plot(centroid(1,:),centroid(2,:),'o','Linewidth',1.5);
disp(strcat('The centroid is (x,y):',num2str(centroid(1,1)),' , ',
    num2str(centroid(2,1))))

title('The paths of the swarm members')
xlabel('x axis','Linewidth',1.25);
ylabel('y axis','Linewidth',1.25);
```

References

[1] Veysel Gazi and Kevin Passino. *Swarm Stability and Optimization*. 01 2011. ISBN 978-3-642-18040-8. doi: 10.1007/978-3-642-18041-5.