Computational Materials Engineering Lab

Experiment 7

Kamesh K (MM16B107)

24 October 2018

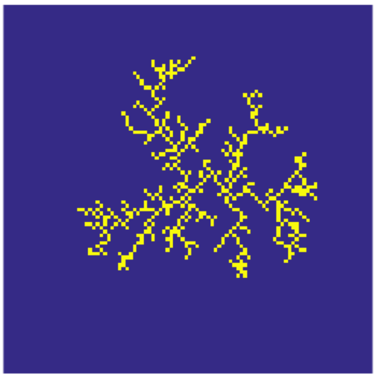
Code for the Isotropic Attachment and Anisotropic Attachment

```
clear all
N=100;  % Computational Grid Size
structure=zeros(N,N);   % Array to store the occupation of particles
Circle_X=[];    % Array to store the X-Coordinates of points inside Annular
Circle_Y=[];    % Array to store the Y-Coordinates of points inside Annular
Circle_Size=0;  % Variable to store number of points inside the Annular
% Loop to obtain the points inside the annular disc
for i=1:N
    for j=1:N
        % Inner radius = 40 and outer radious = 30
        % Can be varied to obtain better outputs
        if abs((i-(N/2))^2+(j-(N/2))^2)<=40*40 && abs((i-(N/2))^2+(j-(N/2))^2)>=30*30
            Circle_X=[Circle_X i];  % Storing the X-Coordinate
            Circle_Y=[Circle_Y j];  % Storing the Y-Coordinate
            Circle_Size=Circle_Size+1;
        end
    end
end
structure(N/2,N/2)=1;   % Initial condition where the center is occupied
hit=1;  % Variable to signify whether it has hit another particle or not
        % hit = 0 implies it has hit, hit = 1 implies not
count=0;% Count of particles inside the domain
Itr_Site=1;             % Iterator for random site
```

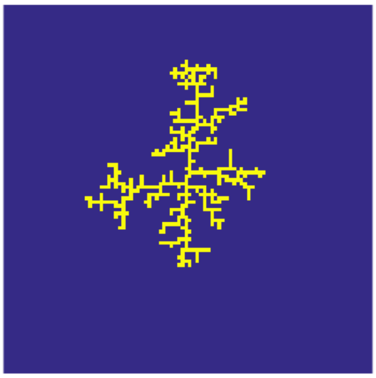
```
Random_Site=randi(Circle_Size,100000,1); % Array storing the random
    numbers
Itr_Move=1;              % Iterator for random movement
Random_Move=randi(8,100000,1);  % Array storing the random movements
while count<500
    % Loop to run till the number of particles forming the structure =
        500
    hit=1;
    Site=Random_Site(Itr_Site); % Random Site Assigned
    Itr_Site=Itr_Site+1;
    if(Itr_Site==100000)
        Itr_Site=1;
        Random_Site=randi(Circle_Size,100000,1); % Reassigning once
            used
    end
    Point_X=Circle_X(Site); % Coordinates of assigned sites
    Point_Y=Circle_Y(Site);
    while hit
    % Looped till the particle hits another particle
    Itr_Move=Itr_Move+1;
    if(Itr_Move==100000)
        Itr_Move=1;
        Random_Move=randi(8,100000,1);  % Reassigning once used
    end
    k=Random_Move(Itr_Move);            % Random movement value
    % Depending on Value some direction is choosen using switch out of
        8
    switch k
    % Respective change in coordinates are carried out based on the
        value
        case 1
            Point_X=Point_X-1;
            Point_Y=Point_Y-1;
        case 2
            Point_X=Point_X;
            Point_Y=Point_Y-1;
        case 3
            Point_X=Point_X+1;
            Point_Y=Point_Y-1;
        case 4
            Point_X=Point_X-1;
            Point_Y=Point_Y;
```

```
            case 5
                Point_X=Point_X+1;
                Point_Y=Point_Y;
            case 6
                Point_X=Point_X-1;
                Point_Y=Point_Y+1;
            case 7
                Point_X=Point_X;
                Point_Y=Point_Y+1;
            case 8
                Point_X=Point_X+1;
                Point_Y=Point_Y+1;
        end
        % Checking whether the particle is still inside the outer circle
        if(abs((Point_X-(N/2))^2+(Point_Y-(N/2))^2)>=40*40)
            break;
        end
        % Checking the Nearest Neighbours available
        % Anisotropy can be achieved by replacing the condition to
        % structure(Point_X,max(Point_Y-1,1))+structure(max(Point_X-1,1),
          Point_Y)+structure(min(Point_X+1,N),Point_Y)+structure(Point_X,
          min(Point_Y+1,N))==1
        if  structure(max(Point_X-1,1),max(Point_Y-1,1)) || structure(
          Point_X,max(Point_Y-1,1)) ||structure(min(Point_X+1,N),max(
          Point_Y-1,1)) ||structure(max(Point_X-1,1),Point_Y) ||structure(
          min(Point_X+1,N),Point_Y) ||structure(max(Point_X-1,1),min(
          Point_Y+1,N)) ||structure(Point_X,min(Point_Y+1,N)) ||structure(
          min(Point_X+1,N),min(Point_Y+1,N))
        % If neighbours are found, we assign the final position
        % Hit is assigned 0 as neighbours are found
                hit=0;
                count=count+1;
                structure(Point_X,Point_Y)=1;
                break;
        end
        hit=1;
        end
    end
    imagesc(structure);
    axis 'square';
    axis off;
```

Output Images



(a) Isotropic Attachment



(b) Anisotropic Attachment

Figure 1: Based on the Attachment

And hence from the figures, its clearly visible that the arrangement of particles are majorly dependent on the nature of the system i.e. Isotropic or Anisotropic.
In case of Isotropic Attachment, the structure is more branched and wide. Whereas in case of Anisotropic Attachment, the structure is confined in few directions to grow and hence we can see a difference in their final structures.