clc;

clear all;

close all;

addpath('DNN')

[filename, pathname] = uigetfile('./ImageDataset/\*.\*');

answer = inputdlg('Enter the noise level(15%,25%,50%) : ','Noise Level');

if strcmp(answer,'15') || strcmp(answer,'15%')

noise = 15;

sigma1 = (noise/255)^2;

elseif strcmp(answer,'25') || strcmp(answer,'25%')

noise = 25;

sigma1 = (noise/255)^2;

elseif strcmp(answer,'50') || strcmp(answer,'50%')

noise = 50;

sigma1 =(noise/255)^2;

else

noise = 15;

sigma1 = (noise/255)^2;

end

I = imread([pathname filename]);

I = imresize(I,[256 256]);

if size(I,3) > 1

I = rgb2gray(I);

else

I = I;

end

noisyI = imnoise(I,'gaussian',0,sigma1);

figure

imshowpair(I,noisyI,'montage');

title('Original Image (left) and Noisy Image (right)')

%

Pop = rand(5,2);

SearchAgents\_no = 5;

Max\_iter = 1;

lb = 0;

ub = 1;

dim = 2;

% initialize position vector and score for the leader

Leader\_pos=zeros(1,dim);

Leader\_score=inf; %change this to -inf for maximization problems

%Initialize the positions of search agents

Positions=initialization(SearchAgents\_no,dim,ub,lb);

Convergence\_curve=zeros(1,Max\_iter);

t=0;% Loop counter

% Main loop

while t<Max\_iter

%% Calculate fitness Function

for i=1:size(Positions,1)

% Return back the search agents that go beyond the boundaries of the search space

Flag4ub=Positions(i,:)>ub;

Flag4lb=Positions(i,:)<lb;

Positions(i,:)=(Positions(i,:).\*(~(Flag4ub+Flag4lb)))+ub.\*Flag4ub+lb.\*Flag4lb;

% Calculate objective function for each search agent

fitness=fitnessfunction(noisyI,I,Positions(i,:));

% Update the leader

if fitness<Leader\_score % Change this to > for maximization problem

Leader\_score=fitness; % Update alpha

Leader\_pos=Positions(i,:);

end

end

fmax = max(fitness);

fmin = min(fitness);

a=2-t\*((2)/Max\_iter); % a decreases linearly fron 2 to 0 in Eq. (2.3)

% a2 linearly dicreases from -1 to -2 to calculate t in Eq. (3.12)

a2=-1+t\*((-1)/Max\_iter);

% Update the Position of search agents

for i=1:size(Positions,1)

r1=(fmax+fmin)/2; % r1 is a random number in [0,1]

r2=(fmax-fmin)/2; % r2 is a random number in [0,1]

A=2\*a\*r1-a; % Eq. (2.3) in the paper

C=2\*r2; % Eq. (2.4) in the paper

b=1; % parameters in Eq. (2.5)

l=(a2-1)\*rand+1; % parameters in Eq. (2.5)

p = rand(); % p in Eq. (2.6)

\*\*\*\*\*(code for position update)

end

t=t+1;

Convergence\_curve(t)=Leader\_score;

[t Leader\_score]

end