Appendix II

Software

Main Program:

```
function varargout = amira_main(varargin)
% AMIRA_MAIN MATLAB code for amira_main.fig
%      AMIRA_MAIN, by itself, creates a new AMIRA_MAIN or raises the existing
%      singleton*.
%
%      H = AMIRA_MAIN returns the handle to a new AMIRA_MAIN or the handle to
%      the existing singleton*.
%
%      AMIRA_MAIN('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in AMIRA_MAIN.M with the given input arguments.
%
%      AMIRA_MAIN('Property','Value',...) creates a new AMIRA_MAIN or raises
the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before amira_main_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to amira_main_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help amira_main

% Last Modified by GUIDE v2.5 08-Mar-2016 12:12:00

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @amira_main_OpeningFcn, ...
                   'gui_OutputFcn',  @amira_main_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
```

```
end
% End initialization code - DO NOT EDIT


% --- Executes just before amira_main is made visible.
function amira_main_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to amira_main (see VARARGIN)

% Choose default command line output for amira_main
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes amira_main wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = amira_main_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
main_algorithm
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

FLD

```
%choosing block of image =

%THIS PROGRAM IS WITHOUT CROPPING THE ORIGINAL IMAGE
clear all
close all
clc
%diary arun.dat on

nop=6%input('number of persons');
nface=3;%input('number of expressions');
%nori=input('number of expressions ');
bs=48;%input('enter block size');
to=0;

for nnp=1:nop%all
if nnp==1%1111111111111111111111111111111111111

for face=1:nface%10   %anger
if face==1
I=imread('KA.AN1.39.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==2
I=imread('KA.AN2.40.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==3
I=imread('KA.AN3.41.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end

s=size(I);
e1=floor(s(1,1)/bs);
e2=floor(s(1,2)/bs);
bl=0;



for a=1:e1 %1
    for b=1:e2 %2
 ro=1;
 co=1;
        for m=bs*a-(bs-1):bs*a%3
               co=0;
            for n=bs*b-(bs-1):bs*b  %4
```

```
                co=co+1;
              bb(ro,co)=I(m,n) ;
              end%4
              ro=ro+1 ;
          end%3

       z=double(bb);
       z=z*z';
       %correlation
      % pause
       %find eigen values
       [w,be]=eig(z);
       diagonal_values=diag(be);
       %sort the eigen values
       [p,k]=sort(diagonal_values);
       %choose the column vector with maximum eigen values
     f=k(bs);
       ev=w(:,f);
    %  pause
      to=to+1;
      valu1(to,:)=ev';
end%2
end %1



end%10



save ang-WITHOUT.mat valu1
save ang-WITHOUT.txt valu1 -ascii
end



%1111111111111111111111111111111111111


%22222222222222222222222222222222
to=0;
if nnp==2

for face=1:nface%10 %disgust
if face==1
I=imread('KA.DI1.42.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
```

```
if face==2
I=imread('KA.DI2.43.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==3
I=imread('KA.DI3.44.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
s=size(I);
e1=floor(s(1,1)/bs);
e2=floor(s(1,2)/bs);
bl=0;



for a=1:e1 %1
    for b=1:e2 %2
 ro=1;
 co=1;
        for m=bs*a-(bs-1):bs*a%3
               co=0;
            for n=bs*b-(bs-1):bs*b  %4
                co=co+1;
            bb(ro,co)=I(m,n) ;
            end%4
            ro=ro+1 ;
        end%3

     z=double(bb);
     z=z*z';
    % pause
     %find eigen values
     [w,be]=eig(z);
     diagonal_values=diag(be);
     %sort the eigen values
     [p,k]=sort(diagonal_values);
     %choose the column vector with maximum eigen values
   f=k(bs);
     ev=w(:,f);
  %  pause
    to=to+1;
    valu2(to,:)=ev';
end%2
end %1



end%10
```

```
save disgust-WITHOUT.mat valu2
    save disgust-WITHOUT.txt valu2 -ascii

end
%22222222222222222222222222222222



%33333333333333333333333333333333333

to=0;
if nnp==3

for face=1:nface%10 %FEAR
if face==1
I=imread('KA.FE1.45.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==2
I=imread('KA.FE2.46.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==3
I=imread('KA.FE3.47.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
s=size(I);
e1=floor(s(1,1)/bs);
e2=floor(s(1,2)/bs);
bl=0;



for a=1:e1 %1
    for b=1:e2 %2
 ro=1;
 co=1;
        for m=bs*a-(bs-1):bs*a%3
               co=0;
            for n=bs*b-(bs-1):bs*b  %4
                co=co+1;
            bb(ro,co)=I(m,n) ;
            end%4
            ro=ro+1 ;
```

```
        end%3

     z=double(bb);
     z=z*z';
    % pause
     %find eigen values
     [w,be]=eig(z);
     diagonal_values=diag(be);
     %sort the eigen values
     [p,k]=sort(diagonal_values);
     %choose the column vector with maximum eigen values
   f=k(bs);
     ev=w(:,f);
  %  pause
    to=to+1;
    valu3(to,:)=ev';
end%2
end %1
end%10

save fear-WITHOUT.mat valu3
save fear-WITHOUT.txt valu3 -ascii
end
%33333333333333333333333333333333333

%444444444444444444444444444444
to=0;
if nnp==4

for face=1:nface%10 %FEAR
if face==1
I=imread('KA.HA1.29.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==2
I=imread('KA.HA2.30.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==3
I=imread('KA.HA3.31.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
s=size(I);
e1=floor(s(1,1)/bs);
e2=floor(s(1,2)/bs);
bl=0;
```

```
for a=1:e1 %1
    for b=1:e2 %2
 ro=1;
 co=1;
        for m=bs*a-(bs-1):bs*a%3
               co=0;
            for n=bs*b-(bs-1):bs*b  %4
                co=co+1;
            bb(ro,co)=I(m,n) ;
            end%4
            ro=ro+1 ;
        end%3

     z=double(bb);
     z=z*z';
    % pause
     %find eigen values
     [w,be]=eig(z);
     diagonal_values=diag(be);
     %sort the eigen values
     [p,k]=sort(diagonal_values);
     %choose the column vector with maximum eigen values
   f=k(bs);
     ev=w(:,f);
  %  pause
    to=to+1;
    valu4(to,:)=ev';
end%2
end %1
end%10

save happy-WITHOUT.mat valu4
save happy-WITHOUT.txt valu4 -ascii
end




%4444444444444444444444444444

%55555555555555555555555555555
to=0;
if nnp==5

for face=1:nface%10 %FEAR
if face==1
I=imread('KA.SA1.33.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
```

```
end
if face==2
I=imread('KA.SA2.34.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==3
I=imread('KA.SA3.35.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
s=size(I);
e1=floor(s(1,1)/bs);
e2=floor(s(1,2)/bs);
bl=0;



for a=1:e1 %1
    for b=1:e2 %2
 ro=1;
 co=1;
        for m=bs*a-(bs-1):bs*a%3
               co=0;
            for n=bs*b-(bs-1):bs*b  %4
                co=co+1;
            bb(ro,co)=I(m,n) ;
            end%4
            ro=ro+1 ;
        end%3

     z=double(bb);
     z=z*z';
    % pause
     %find eigen values
     [w,be]=eig(z);
     diagonal_values=diag(be);
     %sort the eigen values
     [p,k]=sort(diagonal_values);
     %choose the column vector with maximum eigen values
   f=k(bs);
     ev=w(:,f);
  %  pause
    to=to+1;
    valu5(to,:)=ev';
end%2
end %1
end%10

save sad-WITHOUT.mat valu5
save sad-WITHOUT.txt valu5 -ascii
```

```
end



%555555555555555555555555555555
%666666666666666666666666666
to=0;
if nnp==6

for face=1:nface%10 %FEAR
if face==1
I=imread('KA.SU1.36.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==2
I=imread('KA.SU2.37.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==3
I=imread('KA.SU3.38.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
s=size(I);
e1=floor(s(1,1)/bs);
e2=floor(s(1,2)/bs);
bl=0;



for a=1:e1 %1
    for b=1:e2 %2
 ro=1;
 co=1;
        for m=bs*a-(bs-1):bs*a%3
               co=0;
            for n=bs*b-(bs-1):bs*b  %4
                co=co+1;
            bb(ro,co)=I(m,n) ;
            end%4
            ro=ro+1 ;
        end%3

     z=double(bb);
     z=z*z';
    % pause
     %find eigen values
```

```
      [w,be]=eig(z);
      diagonal_values=diag(be);
      %sort the eigen values
      [p,k]=sort(diagonal_values);
      %choose the column vector with maximum eigen values
    f=k(bs);
      ev=w(:,f);
   %  pause
     to=to+1;
     valu6(to,:)=ev';
end%2
end %1
end%10

save sur-WITHOUT.mat valu6
save sur-WITHOUT.txt valu6 -ascii
end




%6666666666666666666666666666

end%all



m1=mean(valu1);
m2=mean(valu2);
m3=mean(valu3);
m4=mean(valu4);
m5=mean(valu5);
m6=mean(valu6);




% if nop==3
% m3=mean(valu3);
% per3=valu3;
% end
% if nop<3
%     m3=0;
% end
mo=(m1+m2+m3+m4+m5+m6)/nop;

per1=valu1;
per2=valu2;
per3=valu3;
```

```
per4=valu4;
per5=valu5;
per6=valu6;
%within class matric
    addsum1=0;
    for q=1:size(per1,1)%person 1
        d=per1(q,:)-m1;
        addsum1=addsum1+d'*d;
    end
    addsum1=addsum1/size(per1,1);

    addsum2=0;
    for q=1:size(per2,1)%person 1
        d=per2(q,:)-m2;
        addsum2=addsum2+d'*d;
    end
    addsum2=addsum2/size(per2,1);




    addsum3=0;
    for q=1:size(per3,1)%person 1
        d=per3(q,:)-m3;
        addsum3=addsum3+d'*d;
    end
    addsum3=addsum3/size(per3,1);

    addsum4=0;
    for q=1:size(per4,1)%person 1
        d=per4(q,:)-m4;
        addsum4=addsum4+d'*d;
    end
    addsum4=addsum4/size(per4,1);



     addsum5=0;
    for q=1:size(per5,1)%person 5
        d=per5(q,:)-m5;
        addsum5=addsum5+d'*d;
    end
    addsum5=addsum5/size(per5,1);


    addsum6=0;
    for q=1:size(per6,1)%person 5
        d=per6(q,:)-m6;
        addsum6=addsum6+d'*d;
```

```
  end
    addsum6=addsum6/size(per6,1);




    sw=(addsum1+addsum2+addsum3+addsum4+addsum5+addsum6)/6;
% end


% if nop<3
%  sw=(addsum1+addsum2)/2;
% end

%between class matrix


    sb=0
    for q=1:size(per1,1)%person 1
        d=per1(q,:)-mo;
        sb=sb+d'*d;
    end

    for q=1:size(per2,1)%person 2
        d=per2(q,:)-mo;
        sb=sb+d'*d;
    end

%     if nop==3
    for q=1:size(per3,1)%person 3
        d=per3(q,:)-mo;
        sb=sb+d'*d;
    end

    for q=1:size(per4,1)%person 4
        d=per4(q,:)-mo;
        sb=sb+d'*d;
    end

    for q=1:size(per5,1)%person 5
        d=per5(q,:)-mo;
        sb=sb+d'*d;
    end

    for q=1:size(per6,1)%person 6
        d=per6(q,:)-mo;
        sb=sb+d'*d;
    end
```

```
sb=sb/(size(per1,1)+size(per2,1)+size(per3,1)+size(per4,1)+size(per5,1)+size(p
er6,1));
% end
% if nop==2
%         sb=sb/(size(per1,1)+size(per2,1));
% end
%singular value decompositon

if(det(sw)==0)

[U,S,V]=svd(sw)

    for t=1:bs
        if(S(t,t)<0.01)
            S(t,t)=.01;   %singularity is eliminated
        end
    end

    sw=U*S*V';
end



%find phi-1 vector


            d=sb*inv(sw);
            [w,be]=eig(d);
             diagonal_values=diag(be);
             %sort the eigen values
             [p,k]=sort(diagonal_values);
             %choose the column vector with maximum eigen values
           f=k(bs);
             phi_1=w(:,f);



%find phi-2 vector
        t1=phi_1*phi_1'*inv(sw);
        t2=phi_1'*inv(sw)*phi_1;
        t3=t1/t2;
        Q=eye(bs)-t3;
        d=Q*sb*inv(sw);
        [w,be]=eig(d);
        diagonal_values=diag(be);
         %sort the eigen values
         [p,k]=sort(diagonal_values);
```

```
          %choose the column vector with maximum eigen values
        f=k(bs);
          phi_2=w(:,f);
%converting into two dimension
          %condition checking
    phy=horzcat(phi_1,phi_2);
    %reducing from higer dimension to two dimension
    two_1=per1*phy;%first person
    two_2=per2*phy;%second person
%
%     if nop==2
%
%
%     plot(two_1(:,1),two_1(:,2),'*',two_2(:,1),two_2(:,2),'+b')
%  legend('Anger','Disgust')
% end

%     if nop==3
    two_3=per3*phy;%third person
    two_4=per4*phy;%fourth person
    two_5=per5*phy;%five person
    two_6=per6*phy;%sixth person

plot(two_1(:,1),two_1(:,2),'*',two_2(:,1),two_2(:,2),'+b',two_3(:,1),two_3(:,2
),'^g',two_4(:,1),two_4(:,2),'*g',two_5(:,1),two_5(:,2),'^r',two_6(:,1),two_6(
:,2),'^k');
    legend('Anger','Disgust','Fear','Happy','Sad','Surprise')
    xlabel('Phi-1 vector')
    ylabel('Phi-2 vector')
% end
%make madifications
Input_Pattern=[two_1;two_2;two_3;two_4;two_5;two_6];
target_pattern=[0.1;0.1;0.1;0.2;0.2;0.2;0.3;0.3;0.3;0.4;0.4;0.4;0.5;0.5;0.5;0.
6;0.6;0.6];
save fweight.mat  phi_1 phi_2 Input_Pattern target_pattern
%diary off
```

FLD-BPA

```
choosing block of image =

%THIS PROGRAM IS WITHOUT CROPPING THE ORIGINAL IMAGE
clear all
close all
clc
%diary arun.dat on

nop=6%input('number of persons');
nface=3;%input('number of expressions');
%nori=input('number of expressions ');
bs=input('enter block size');
to=0;

for nnp=1:nop%all
if nnp==1%1111111111111111111111111111111111111

for face=1:nface%10   %anger
if face==1
I=imread('KA.AN1.39.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==2
I=imread('KA.AN2.40.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==3
I=imread('KA.AN3.41.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end

s=size(I);
e1=floor(s(1,1)/bs);
e2=floor(s(1,2)/bs);
bl=0;



for a=1:e1 %1
    for b=1:e2 %2
 ro=1;
 co=1;
        for m=bs*a-(bs-1):bs*a%3
               co=0;
            for n=bs*b-(bs-1):bs*b  %4
```

```
                co=co+1;
              bb(ro,co)=I(m,n) ;
              end%4
              ro=ro+1 ;
          end%3

       z=double(bb);
       z=z*z';
       %correlation
      % pause
       %find eigen values
       [w,be]=eig(z);
       diagonal_values=diag(be);
       %sort the eigen values
       [p,k]=sort(diagonal_values);
       %choose the column vector with maximum eigen values
     f=k(bs);
       ev=w(:,f);
    %  pause
      to=to+1;
      valu1(to,:)=ev';
end%2
end %1



end%10



save ang-WITHOUT.mat valu1
save ang-WITHOUT.txt valu1 -ascii
end



%1111111111111111111111111111111111111


%22222222222222222222222222222222
to=0;
if nnp==2

for face=1:nface%10 %disgust
if face==1
I=imread('KA.DI1.42.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
```

```
if face==2
I=imread('KA.DI2.43.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==3
I=imread('KA.DI3.44.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
s=size(I);
e1=floor(s(1,1)/bs);
e2=floor(s(1,2)/bs);
bl=0;



for a=1:e1 %1
    for b=1:e2 %2
 ro=1;
 co=1;
        for m=bs*a-(bs-1):bs*a%3
               co=0;
            for n=bs*b-(bs-1):bs*b  %4
                co=co+1;
            bb(ro,co)=I(m,n) ;
            end%4
            ro=ro+1 ;
        end%3

     z=double(bb);
     z=z*z';
    % pause
     %find eigen values
     [w,be]=eig(z);
     diagonal_values=diag(be);
     %sort the eigen values
     [p,k]=sort(diagonal_values);
     %choose the column vector with maximum eigen values
   f=k(bs);
     ev=w(:,f);
  %  pause
    to=to+1;
    valu2(to,:)=ev';
end%2
end %1



end%10
```

```
save disgust-WITHOUT.mat valu2
    save disgust-WITHOUT.txt valu2 -ascii

end
%22222222222222222222222222222222



%33333333333333333333333333333333333

to=0;
if nnp==3

for face=1:nface%10 %FEAR
if face==1
I=imread('KA.FE1.45.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==2
I=imread('KA.FE2.46.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==3
I=imread('KA.FE3.47.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
s=size(I);
e1=floor(s(1,1)/bs);
e2=floor(s(1,2)/bs);
bl=0;



for a=1:e1 %1
    for b=1:e2 %2
 ro=1;
 co=1;
        for m=bs*a-(bs-1):bs*a%3
               co=0;
            for n=bs*b-(bs-1):bs*b  %4
                co=co+1;
            bb(ro,co)=I(m,n) ;
            end%4
            ro=ro+1 ;
```

```
        end%3

     z=double(bb);
     z=z*z';
    % pause
     %find eigen values
     [w,be]=eig(z);
     diagonal_values=diag(be);
     %sort the eigen values
     [p,k]=sort(diagonal_values);
     %choose the column vector with maximum eigen values
   f=k(bs);
     ev=w(:,f);
  %  pause
    to=to+1;
    valu3(to,:)=ev';
end%2
end %1
end%10

save fear-WITHOUT.mat valu3
save fear-WITHOUT.txt valu3 -ascii
end
%33333333333333333333333333333333333

%444444444444444444444444444444
to=0;
if nnp==4

for face=1:nface%10 %FEAR
if face==1
I=imread('KA.HA1.29.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==2
I=imread('KA.HA2.30.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==3
I=imread('KA.HA3.31.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
s=size(I);
e1=floor(s(1,1)/bs);
e2=floor(s(1,2)/bs);
bl=0;
```

```
for a=1:e1 %1
    for b=1:e2 %2
 ro=1;
 co=1;
        for m=bs*a-(bs-1):bs*a%3
               co=0;
            for n=bs*b-(bs-1):bs*b  %4
                co=co+1;
            bb(ro,co)=I(m,n) ;
            end%4
            ro=ro+1 ;
        end%3

     z=double(bb);
     z=z*z';
    % pause
     %find eigen values
     [w,be]=eig(z);
     diagonal_values=diag(be);
     %sort the eigen values
     [p,k]=sort(diagonal_values);
     %choose the column vector with maximum eigen values
   f=k(bs);
     ev=w(:,f);
  %  pause
    to=to+1;
    valu4(to,:)=ev';
end%2
end %1
end%10

save happy-WITHOUT.mat valu4
save happy-WITHOUT.txt valu4 -ascii
end




%4444444444444444444444444444

%55555555555555555555555555555
to=0;
if nnp==5

for face=1:nface%10 %FEAR
if face==1
I=imread('KA.SA1.33.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
```

```
end
if face==2
I=imread('KA.SA2.34.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==3
I=imread('KA.SA3.35.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
s=size(I);
e1=floor(s(1,1)/bs);
e2=floor(s(1,2)/bs);
bl=0;



for a=1:e1 %1
    for b=1:e2 %2
 ro=1;
 co=1;
        for m=bs*a-(bs-1):bs*a%3
               co=0;
            for n=bs*b-(bs-1):bs*b  %4
                co=co+1;
            bb(ro,co)=I(m,n) ;
            end%4
            ro=ro+1 ;
        end%3

     z=double(bb);
     z=z*z';
    % pause
     %find eigen values
     [w,be]=eig(z);
     diagonal_values=diag(be);
     %sort the eigen values
     [p,k]=sort(diagonal_values);
     %choose the column vector with maximum eigen values
   f=k(bs);
     ev=w(:,f);
  %  pause
    to=to+1;
    valu5(to,:)=ev';
end%2
end %1
end%10

save sad-WITHOUT.mat valu5
save sad-WITHOUT.txt valu5 -ascii
```

```
end



%555555555555555555555555555555
%666666666666666666666666666
to=0;
if nnp==6

for face=1:nface%10 %FEAR
if face==1
I=imread('KA.SU1.36.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==2
I=imread('KA.SU2.37.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==3
I=imread('KA.SU3.38.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
s=size(I);
e1=floor(s(1,1)/bs);
e2=floor(s(1,2)/bs);
bl=0;



for a=1:e1 %1
    for b=1:e2 %2
 ro=1;
 co=1;
        for m=bs*a-(bs-1):bs*a%3
               co=0;
            for n=bs*b-(bs-1):bs*b  %4
                co=co+1;
            bb(ro,co)=I(m,n) ;
            end%4
            ro=ro+1 ;
        end%3

     z=double(bb);
     z=z*z';
    % pause
     %find eigen values
```

```
      [w,be]=eig(z);
      diagonal_values=diag(be);
      %sort the eigen values
      [p,k]=sort(diagonal_values);
      %choose the column vector with maximum eigen values
    f=k(bs);
      ev=w(:,f);
   %  pause
     to=to+1;
     valu6(to,:)=ev';
end%2
end %1
end%10

save sur-WITHOUT.mat valu6
save sur-WITHOUT.txt valu6 -ascii
end




%6666666666666666666666666666

end%all



m1=mean(valu1);
m2=mean(valu2);
m3=mean(valu3);
m4=mean(valu4);
m5=mean(valu5);
m6=mean(valu6);




% if nop==3
% m3=mean(valu3);
% per3=valu3;
% end
% if nop<3
%     m3=0;
% end
mo=(m1+m2+m3+m4+m5+m6)/nop;

per1=valu1;
per2=valu2;
per3=valu3;
```

```
per4=valu4;
per5=valu5;
per6=valu6;
%within class matric
    addsum1=0;
    for q=1:size(per1,1)%person 1
        d=per1(q,:)-m1;
        addsum1=addsum1+d'*d;
    end
    addsum1=addsum1/size(per1,1);

    addsum2=0;
    for q=1:size(per2,1)%person 1
        d=per2(q,:)-m2;
        addsum2=addsum2+d'*d;
    end
    addsum2=addsum2/size(per2,1);









    addsum3=0;
    for q=1:size(per3,1)%person 1
        d=per3(q,:)-m3;
        addsum3=addsum3+d'*d;
    end
    addsum3=addsum3/size(per3,1);

    addsum4=0;
    for q=1:size(per4,1)%person 1
        d=per4(q,:)-m4;
        addsum4=addsum4+d'*d;
    end
    addsum4=addsum4/size(per4,1);



     addsum5=0;
    for q=1:size(per5,1)%person 5
        d=per5(q,:)-m5;
        addsum5=addsum5+d'*d;
    end
    addsum5=addsum5/size(per5,1);


    addsum6=0;
    for q=1:size(per6,1)%person 5
```

```
        d=per6(q,:)-m6;
        addsum6=addsum6+d'*d;
  end
    addsum6=addsum6/size(per6,1);




    sw=(addsum1+addsum2+addsum3+addsum4+addsum5+addsum6)/6;
% end


% if nop<3
%  sw=(addsum1+addsum2)/2;
% end

%between class matrix


    sb=0
    for q=1:size(per1,1)%person 1
        d=per1(q,:)-mo;
        sb=sb+d'*d;
    end

    for q=1:size(per2,1)%person 2
        d=per2(q,:)-mo;
        sb=sb+d'*d;
    end

%     if nop==3
    for q=1:size(per3,1)%person 3
        d=per3(q,:)-mo;
        sb=sb+d'*d;
    end

    for q=1:size(per4,1)%person 4
        d=per4(q,:)-mo;
        sb=sb+d'*d;
    end

    for q=1:size(per5,1)%person 5
        d=per5(q,:)-mo;
        sb=sb+d'*d;
    end

    for q=1:size(per6,1)%person 6
        d=per6(q,:)-mo;
        sb=sb+d'*d;
```

```
    end


sb=sb/(size(per1,1)+size(per2,1)+size(per3,1)+size(per4,1)+size(per5,1)+size(p
er6,1));
% end
% if nop==2
%         sb=sb/(size(per1,1)+size(per2,1));
% end
%singular value decompositon

if(det(sw)==0)

[U,S,V]=svd(sw)

    for t=1:bs
        if(S(t,t)<0.01)
            S(t,t)=.01;   %singularity is eliminated
        end
    end

    sw=U*S*V';
end



%find phi-1 vector


            d=sb*inv(sw);
            [w,be]=eig(d);
             diagonal_values=diag(be);
             %sort the eigen values
             [p,k]=sort(diagonal_values);
             %choose the column vector with maximum eigen values
           f=k(bs);
             phi_1=w(:,f);



%find phi-2 vector
        t1=phi_1*phi_1'*inv(sw);
        t2=phi_1'*inv(sw)*phi_1;
        t3=t1/t2;
        Q=eye(bs)-t3;
        d=Q*sb*inv(sw);
        [w,be]=eig(d);
        diagonal_values=diag(be);
```

```
          %sort the eigen values
          [p,k]=sort(diagonal_values);
          %choose the column vector with maximum eigen values
        f=k(bs);
          phi_2=w(:,f);
%converting into two dimension
          %condition checking
    phy=horzcat(phi_1,phi_2);
    %reducing from higer dimension to two dimension
    two_1=per1*phy;%first person
    two_2=per2*phy;%second person
%
%     if nop==2
%
%
%     plot(two_1(:,1),two_1(:,2),'*',two_2(:,1),two_2(:,2),'+b')
%  legend('Anger','Disgust')
% end

%     if nop==3
    two_3=per3*phy;%third person
    two_4=per4*phy;%fourth person
    two_5=per5*phy;%five person
    two_6=per6*phy;%sixth person

plot(two_1(:,1),two_1(:,2),'*',two_2(:,1),two_2(:,2),'+b',two_3(:,1),two_3(:,2
),'^g',two_4(:,1),two_4(:,2),'*g',two_5(:,1),two_5(:,2),'^r',two_6(:,1),two_6(
:,2),'^k');
    legend('Anger','Disgust','Fear','Happy','Sad','Surprise')
    xlabel('Phi-1 vector')
    ylabel('Phi-2 vector')
% end
%make madifications
Input_Pattern=[two_1;two_2;two_3;two_4;two_5;two_6];
target_pattern=[0.1;0.1;0.1;0.2;0.2;0.2;0.3;0.3;0.3;0.4;0.4;0.4;0.5;0.5;0.5;0.
6;0.6;0.6];
save fweight_only_fld.mat  phi_1 phi_2 Input_Pattern target_pattern
%diary off
```

FLD-SLN

```
%choosing block of image =

%THIS PROGRAM IS WITHOUT CROPPING THE ORIGINAL IMAGE
clear all
close all
clc
%diary arun.dat on

nop=6%input('number of persons');
nface=3;%input('number of expressions');
%nori=input('number of expressions ');
bs=input('enter block size');
to=0;

for nnp=1:nop%all
if nnp==1%1111111111111111111111111111111111111

for face=1:nface%10   %anger
if face==1
I=imread('KA.AN1.39.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==2
I=imread('KA.AN2.40.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==3
I=imread('KA.AN3.41.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end

s=size(I);
e1=floor(s(1,1)/bs);
e2=floor(s(1,2)/bs);
bl=0;



for a=1:e1 %1
    for b=1:e2 %2
 ro=1;
 co=1;
        for m=bs*a-(bs-1):bs*a%3
               co=0;
            for n=bs*b-(bs-1):bs*b  %4
```

```
                co=co+1;
              bb(ro,co)=I(m,n) ;
              end%4
              ro=ro+1 ;
          end%3

      z=double(bb);
      z=z*z';
      %correlation
     % pause
      %find eigen values
      [w,be]=eig(z);
      diagonal_values=diag(be);
      %sort the eigen values
      [p,k]=sort(diagonal_values);
      %choose the column vector with maximum eigen values
    f=k(bs);
      ev=w(:,f);
   %  pause
     to=to+1;
     valu1(to,:)=ev';
end%2
end %1



end%10



save ang-WITHOUT.mat valu1
save ang-WITHOUT.txt valu1 -ascii
end


%1111111111111111111111111111111111111


%22222222222222222222222222222222
to=0;
if nnp==2

for face=1:nface%10 %disgust
if face==1
I=imread('KA.DI1.42.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==2
I=imread('KA.DI2.43.tiff');
```

```
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==3
I=imread('KA.DI3.44.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
s=size(I);
e1=floor(s(1,1)/bs);
e2=floor(s(1,2)/bs);
bl=0;



for a=1:e1 %1
    for b=1:e2 %2
 ro=1;
 co=1;
        for m=bs*a-(bs-1):bs*a%3
               co=0;
            for n=bs*b-(bs-1):bs*b  %4
                co=co+1;
            bb(ro,co)=I(m,n) ;
            end%4
            ro=ro+1 ;
        end%3

     z=double(bb);
     z=z*z';
    % pause
     %find eigen values
     [w,be]=eig(z);
     diagonal_values=diag(be);
     %sort the eigen values
     [p,k]=sort(diagonal_values);
     %choose the column vector with maximum eigen values
   f=k(bs);
     ev=w(:,f);
  %  pause
    to=to+1;
    valu2(to,:)=ev';
end%2
end %1



end%10
```

```
save disgust-WITHOUT.mat valu2
    save disgust-WITHOUT.txt valu2 -ascii

end
%22222222222222222222222222222222

%33333333333333333333333333333333

to=0;
if nnp==3

for face=1:nface%10 %FEAR
if face==1
I=imread('KA.FE1.45.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==2
I=imread('KA.FE2.46.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==3
I=imread('KA.FE3.47.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
s=size(I);
e1=floor(s(1,1)/bs);
e2=floor(s(1,2)/bs);
bl=0;

for a=1:e1 %1
    for b=1:e2 %2
 ro=1;
 co=1;
        for m=bs*a-(bs-1):bs*a%3
               co=0;
            for n=bs*b-(bs-1):bs*b  %4
                co=co+1;
            bb(ro,co)=I(m,n) ;
            end%4
            ro=ro+1 ;
        end%3
```

```
     z=double(bb);
     z=z*z';
    % pause
     %find eigen values
     [w,be]=eig(z);
     diagonal_values=diag(be);
     %sort the eigen values
     [p,k]=sort(diagonal_values);
     %choose the column vector with maximum eigen values
   f=k(bs);
     ev=w(:,f);
  %  pause
    to=to+1;
    valu3(to,:)=ev';
end%2
end %1
end%10

save fear-WITHOUT.mat valu3
save fear-WITHOUT.txt valu3 -ascii
end
%333333333333333333333333333333333333

%444444444444444444444444444444
to=0;
if nnp==4

for face=1:nface%10 %FEAR
if face==1
I=imread('KA.HA1.29.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==2
I=imread('KA.HA2.30.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==3
I=imread('KA.HA3.31.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
s=size(I);
e1=floor(s(1,1)/bs);
e2=floor(s(1,2)/bs);
bl=0;



for a=1:e1 %1
```

```
    for b=1:e2 %2
 ro=1;
 co=1;
        for m=bs*a-(bs-1):bs*a%3
               co=0;
            for n=bs*b-(bs-1):bs*b  %4
                co=co+1;
            bb(ro,co)=I(m,n) ;
            end%4
            ro=ro+1 ;
        end%3

     z=double(bb);
     z=z*z';
    % pause
     %find eigen values
     [w,be]=eig(z);
     diagonal_values=diag(be);
     %sort the eigen values
     [p,k]=sort(diagonal_values);
     %choose the column vector with maximum eigen values
   f=k(bs);
     ev=w(:,f);
  %  pause
    to=to+1;
    valu4(to,:)=ev';
end%2
end %1
end%10

save happy-WITHOUT.mat valu4
save happy-WITHOUT.txt valu4 -ascii
end




%4444444444444444444444444444

%55555555555555555555555555555
to=0;
if nnp==5

for face=1:nface%10 %FEAR
if face==1
I=imread('KA.SA1.33.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==2
```

```
I=imread('KA.SA2.34.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==3
I=imread('KA.SA3.35.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
s=size(I);
e1=floor(s(1,1)/bs);
e2=floor(s(1,2)/bs);
bl=0;



for a=1:e1 %1
    for b=1:e2 %2
 ro=1;
 co=1;
        for m=bs*a-(bs-1):bs*a%3
               co=0;
            for n=bs*b-(bs-1):bs*b  %4
                co=co+1;
            bb(ro,co)=I(m,n) ;
            end%4
            ro=ro+1 ;
        end%3

     z=double(bb);
     z=z*z';
    % pause
     %find eigen values
     [w,be]=eig(z);
     diagonal_values=diag(be);
     %sort the eigen values
     [p,k]=sort(diagonal_values);
     %choose the column vector with maximum eigen values
   f=k(bs);
     ev=w(:,f);
  %  pause
    to=to+1;
    valu5(to,:)=ev';
end%2
end %1
end%10

save sad-WITHOUT.mat valu5
save sad-WITHOUT.txt valu5 -ascii
end
```

```
%555555555555555555555555555555
%666666666666666666666666666
to=0;
if nnp==6

for face=1:nface%10 %FEAR
if face==1
I=imread('KA.SU1.36.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==2
I=imread('KA.SU2.37.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
if face==3
I=imread('KA.SU3.38.tiff');
% I=rgb2gray(I);
I=I(1:bs,1:bs);
end
s=size(I);
e1=floor(s(1,1)/bs);
e2=floor(s(1,2)/bs);
bl=0;



for a=1:e1 %1
    for b=1:e2 %2
 ro=1;
 co=1;
        for m=bs*a-(bs-1):bs*a%3
               co=0;
            for n=bs*b-(bs-1):bs*b  %4
                co=co+1;
            bb(ro,co)=I(m,n) ;
            end%4
            ro=ro+1 ;
        end%3

     z=double(bb);
     z=z*z';
    % pause
     %find eigen values
     [w,be]=eig(z);
     diagonal_values=diag(be);
```

```
     %sort the eigen values
     [p,k]=sort(diagonal_values);
     %choose the column vector with maximum eigen values
   f=k(bs);
     ev=w(:,f);
  %  pause
    to=to+1;
    valu6(to,:)=ev';
end%2
end %1
end%10

save sur-WITHOUT.mat valu6
save sur-WITHOUT.txt valu6 -ascii
end




%6666666666666666666666666666

end%all



m1=mean(valu1);
m2=mean(valu2);
m3=mean(valu3);
m4=mean(valu4);
m5=mean(valu5);
m6=mean(valu6);




% if nop==3
% m3=mean(valu3);
% per3=valu3;
% end
% if nop<3
%     m3=0;
% end
mo=(m1+m2+m3+m4+m5+m6)/nop;

per1=valu1;
per2=valu2;
per3=valu3;
per4=valu4;
per5=valu5;
```

```
per6=valu6;
%within class matric
    addsum1=0;
    for q=1:size(per1,1)%person 1
        d=per1(q,:)-m1;
        addsum1=addsum1+d'*d;
    end
    addsum1=addsum1/size(per1,1);

    addsum2=0;
    for q=1:size(per2,1)%person 1
        d=per2(q,:)-m2;
        addsum2=addsum2+d'*d;
    end
    addsum2=addsum2/size(per2,1);









    addsum3=0;
    for q=1:size(per3,1)%person 1
        d=per3(q,:)-m3;
        addsum3=addsum3+d'*d;
    end
    addsum3=addsum3/size(per3,1);

    addsum4=0;
    for q=1:size(per4,1)%person 1
        d=per4(q,:)-m4;
        addsum4=addsum4+d'*d;
    end
    addsum4=addsum4/size(per4,1);



     addsum5=0;
    for q=1:size(per5,1)%person 5
        d=per5(q,:)-m5;
        addsum5=addsum5+d'*d;
    end
    addsum5=addsum5/size(per5,1);


    addsum6=0;
    for q=1:size(per6,1)%person 5
        d=per6(q,:)-m6;
        addsum6=addsum6+d'*d;
```

```
  end
    addsum6=addsum6/size(per6,1);




    sw=(addsum1+addsum2+addsum3+addsum4+addsum5+addsum6)/6;
% end


% if nop<3
%  sw=(addsum1+addsum2)/2;
% end

%between class matrix


    sb=0
    for q=1:size(per1,1)%person 1
        d=per1(q,:)-mo;
        sb=sb+d'*d;
    end

    for q=1:size(per2,1)%person 2
        d=per2(q,:)-mo;
        sb=sb+d'*d;
    end

%     if nop==3
    for q=1:size(per3,1)%person 3
        d=per3(q,:)-mo;
        sb=sb+d'*d;
    end

    for q=1:size(per4,1)%person 4
        d=per4(q,:)-mo;
        sb=sb+d'*d;
    end

    for q=1:size(per5,1)%person 5
        d=per5(q,:)-mo;
        sb=sb+d'*d;
    end

    for q=1:size(per6,1)%person 6
        d=per6(q,:)-mo;
        sb=sb+d'*d;
    end
```

```
sb=sb/(size(per1,1)+size(per2,1)+size(per3,1)+size(per4,1)+size(per5,1)+size(p
er6,1));
% end
% if nop==2
%         sb=sb/(size(per1,1)+size(per2,1));
% end
%singular value decompositon

if(det(sw)==0)

[U,S,V]=svd(sw)

    for t=1:bs
        if(S(t,t)<0.01)
            S(t,t)=.01;   %singularity is eliminated
        end
    end

    sw=U*S*V';
end



%find phi-1 vector


            d=sb*inv(sw);
            [w,be]=eig(d);
             diagonal_values=diag(be);
             %sort the eigen values
             [p,k]=sort(diagonal_values);
             %choose the column vector with maximum eigen values
           f=k(bs);
             phi_1=w(:,f);



%find phi-2 vector
        t1=phi_1*phi_1'*inv(sw);
        t2=phi_1'*inv(sw)*phi_1;
        t3=t1/t2;
        Q=eye(bs)-t3;
        d=Q*sb*inv(sw);
        [w,be]=eig(d);
        diagonal_values=diag(be);
         %sort the eigen values
```

```
          [p,k]=sort(diagonal_values);
          %choose the column vector with maximum eigen values
        f=k(bs);
          phi_2=w(:,f);
%converting into two dimension
          %condition checking
    phy=horzcat(phi_1,phi_2);
    %reducing from higer dimension to two dimension
    two_1=per1*phy;%first person
    two_2=per2*phy;%second person
%
%     if nop==2
%
%
%     plot(two_1(:,1),two_1(:,2),'*',two_2(:,1),two_2(:,2),'+b')
%  legend('Anger','Disgust')
% end

%     if nop==3
    two_3=per3*phy;%third person
    two_4=per4*phy;%fourth person
    two_5=per5*phy;%five person
    two_6=per6*phy;%sixth person

plot(two_1(:,1),two_1(:,2),'*',two_2(:,1),two_2(:,2),'+b',two_3(:,1),two_3(:,2
),'^g',two_4(:,1),two_4(:,2),'*g',two_5(:,1),two_5(:,2),'^r',two_6(:,1),two_6(
:,2),'^k');
    legend('Anger','Disgust','Fear','Happy','Sad','Surprise')
    xlabel('Phi-1 vector')
    ylabel('Phi-2 vector')
% end
%make madifications
Input_Pattern=[two_1;two_2;two_3;two_4;two_5;two_6];
target_pattern=[0.1;0.1;0.1;0.2;0.2;0.2;0.3;0.3;0.3;0.4;0.4;0.4;0.5;0.5;0.5;0.
6;0.6;0.6];
save fweight_only_fld.mat  phi_1 phi_2 Input_Pattern target_pattern
%diary off
```

FLD-CMAC

```
%Linear combination
function CMAC_train(inputta,xorout,trpat,tepat)
load before_hid.mat
load before_out.mat
format long

load ra3.f -ascii

hl=input('Number of nodes in hidden layer=');
%hl=5;
 %xorin=X1inputR ;
% xorout=round(X1targetR*3);
xorin=inputta(1:trpat,:);


Cor=xorin;
ss=size(Cor);
nv=ss(1,2);



Act=xorout(1:trpat,:);%/10;
s=size(Act);
np=s(1,1);
nt=s(1,2);
%pause

ol=nt;
il=nv;


%assign weights between input layer and hidden

ijj=1;
for i=1:il%45
%    i
    for j=1:hl%46

        wih(i,j)=ra3(ijj);
            ijj=ijj+1;
end%46
end%45
wih=wih(1:il,1:hl);


ij=1;
```

```
for i=1:hl%47
    for j=1:ol%48
    hou(i,j)=ra3(ij);
    ij=ij+1;
end%48
end%47


hou=hou(1:hl,1:ol);

eta=1;
%MSE=input('Desired Mean squared error');
MSE=0.0000001;



 for ty=1:10000000%outer loop
erp=0;


for we=1:np        %to form a cycle



    for yty=1:nv
    a1(yty)=Cor(we,yty);
end
 for yty=1:nt
    tar(yty)=Act(we,yty);
end






%BPABPABPABPABPABPABPABPA

%transpose a


 %forward operation
        %linear summation to nodes in hidden layer



      pa2=a1*wih;%inputs to nodes in the hidden layer
```

```
      %do quantization from input layer to hidden layer
      for local1=1:size(pa2,2)

          pu1=abs(pa2(local1)-quntbox);
         [y,v]= sort(pu1);
%          M1= min(pu1);
%         Loc1= find(pu1==M1);
%         pa2(local1)=v(1)/10000;
         pa2(local1)=-1*y(1);%/10000;
          clear y v pu1
      end

clear y v pu1

        for y=1:hl %11

            a2(y)= 1/(1+exp(-pa2(y)) );%outputs from nodes in the hidden layer
        end  %11
        %inputs to nodes in the output layer

        pa3=a2*hou;
        %do quantization  for hidden to output layer
        for local1=1:size(pa3,2)

          pu1=abs(pa3(local1)-quntibox);
          [y,v]= sort(pu1);
%          M1= min(pu1);
%         Loc1= find(pu1==M1);
%         pa3(local1)=v(1)/10000;
pa3(local1)=y(1);
            clear y v pu1

      end

    %outputs from nodes in the output layer
        for y=1:ol%12
        a3(y)=1/(1+exp(-pa3(y)));

    end%12

       %Error of pattern calculation

          summm=0;
        for k=1:ol%13
            t=tar(k)-a3(k);
        summm=summm+(t*t)/2;
         end%13

%error of pattern
```

```
  erp=erp+summm;
  % disp('erp')

  %reverse operation
%Calculation of delta in the output layer
 for k=1:ol%14
     t=tar(k)-a3(k);
     t1=1-a3(k);
 deloutput(k)=a3(k)*t1*t;
  end%14


  %updating weights between output layer and hidden layer
  for k=1:hl%16
      for kk=1:ol%15
          hou(k,kk)=hou(k,kk)+eta*deloutput(kk)*a2(k);
      end%15
  end%16



  %calculation of summation for the nodes in the hidden layer

gh=  hou';
  summa=deloutput*gh;
         %Calculation of delta in the hidden layer
 for k=1:hl%17

     t1=1-a2(k);
 delhidden(k)=a2(k)*t1*summa(k);
  end%17

  %weight updation in the input and hidden layer

  for k=1:il%19
      for kk=1:hl%18
          wih(k,kk)=wih(k,kk)+eta*delhidden(kk)*a1(k);
      end%18
  end%19


  %end of reverse



%    disp('MSe')
```

```
        end % %to form a cycle


        if mod(ty,1)==0

% save wih.mat wih -ascii
%     save houL.mat hou -ascii

%ty
erp;


[Classified_percentage]=CMAC_testing_for_all(inputta(1:tepat,:),xorout(1:tepat
,:),wih,hou);%common function
Re(ty,1)=ty;
        Re(ty,2)=erp;
        Re(ty,3)=Classified_percentage;
         save bw1.mat Re wih hou hl  trpat tepat eta
         Re;
         disp('Iterations,  Mean squared error, Percentage classification')
         [ty erp Classified_percentage]
        end

%pause

if erp<MSE | Classified_percentage>74
    save bw1.mat Re wih hou hl trpat tepat eta
    disp('training over')
    break
end

end%outer loop
p1=Re;
ep=Re;
%subplot(1,2,1)
plot(ep(:,1),ep(:,2),'-')
xlabel('Iterations')
ylabel('MSE')

%subplot(1,2,2)
figure
%plot(-.1,-.1,'.',0,100,'.',perc(:,1),perc(:,3)+45,'-')
epp=[p1(:,1),p1(:,3)];
epp=[0 0;epp]
plot(epp(:,1),epp(:,2),'-')
xlabel('Iterations')
ylabel('% Classifications')
```

Gabor

```
clear all
clc

RGB = imread('KA.SU3.38.tiff');
%          [X,map] = rgb2ind(RGB,128);
%          imshow(X,map);
%I=imread('h256.bmp');
%I=imrotate(I,90,'bilinear');
%I=imrotate(I,90);
%II=imrotate(I,90);
X=RGB;
I1=double(X);
%figure,imshow(I1)
I2=I1(1:99,1:99);
%I2=I1;
ss=size(I2);
%figure,imshow(I2)
%(smoothing using gaussian smoothing function along direction angle 0)
s=1;                                                    %scale 1
it=1;
for x=1:ss(1,1)
for y=1:ss(1,2)
    w0(x,y)=(-(x/s)*exp(-(x*x+y*y)/(2*s*s)));
   %c(1,it)=w0(x,y);
   it=it+1;
   end
end
it=1;
fw0=conv2(w0,I2,'full');
fw0=fw0(1:ss(1,1),1:ss(1,2));
fw01=fw0;
bs=9;
nma=fw0;
im=abs(fw0);

e1=ss(1,1)-(bs-1);
e2=ss(1,2)-(bs-1);

de=1;
for a=1:e1   %1

    for b=1:e2  %2


iy=1;
 ro=1;
 co=1;
```

```
 rs=a;
          re=a+(bs-1);
          cs=b;
          ce=b+(bs-1);


 sum=0;
        for m=rs:re %3

            co=0;


            for n=cs:ce  %4
                co=co+1;



                bb(ro,co)=im(m,n) ;
%                sum=sum+bb(ro,co);

     iy=iy+1;

        end %4
            ro=ro+1 ;

        end%3

%sum=sum/(bs*bs);
as=mean(mean(abs(bb)));
            im(a+1,b+1)=as;      %eq(9)
%     ma(de)=max(max( as));
            de=de+1;
    end
end
%/////
%directional feature
%equation 12
imm=im;
  wq=max(imm);

imm1=abs(log(imm));


%/////
%finding total energy  Equation(14)
%s=size(Act);
bs=9%input('Enter block size')

%imm=;
```

```
e1=floor(ss(1,1)/bs);
e2=floor(ss(1,2)/bs);
de=1;
for a=1:e1   %1

    for b=1:e2  %2
iy=1;
 ro=1;
 co=1;

          rs=bs*a-(bs-1);
          re=bs*a;
          cs=bs*b-(bs-1);
          ce=bs*b;

          summ=0;
         for m=rs:re %3

             co=0;
             for n=cs:ce  %4
                 co=co+1;
                 bb(ro,co)=imm(m,n) ;
                 summ=summ+bb(ro,co);
     iy=iy+1;

         end %4
             ro=ro+1 ;

         end%3


sa(a,b)=summ./81;

%sum=sum/(bs*bs);
%as=mean(mean(abs(bb)));
 %            im(a+1,b+1)=as;     %eq(9)
%     ma(de)=max(max( as));
             de=de+1;
    end
end
%/////

%ETot2=sa;

%percentage energy
bs=9%input('Enter block size')

%imm=nma;
e1=floor(ss(1,1)/bs);
e2=floor(ss(1,2)/bs);
```

```
de=1;
for a=1:e1   %1

    for b=1:e2  %2
iy=1;
 ro=1;
 co=1;


          rs=bs*a-(bs-1);
          re=bs*a;
          cs=bs*b-(bs-1);
          ce=bs*b;


          sum=0;
         for m=rs:re %3

             co=0;
             for n=cs:ce  %4
                 co=co+1;
                 pimm(m,n)=imm(m,n)/sa(a,b) ;
     iy=iy+1;

         end %4
             ro=ro+1 ;

         end%3
             de=de+1;
    end
end
%/////
%weighted roughness
  wr01=wq*pimm;



  %############



  %##########
```

```
  %!!!!!!!!!
for x=1:ss(1,1)
for y=1:ss(1,2)
    w90(x,y)=(-(y/s)*exp(-(x*x+y*y)/(2*s*s)));
   %c(1,it)=w0(x,y);
   it=it+1;
   end
end
it=1;
fw90=conv2(w90,I2,'full');
fw90=fw90(1:ss(1,1),1:ss(1,2));
fw901=fw90;
bs=9;
nma=fw90;
im=abs(fw90);

e1=ss(1,1)-(bs-1);
e2=ss(1,2)-(bs-1);

de=1;
for a=1:e1   %1

    for b=1:e2  %2


iy=1;
 ro=1;
 co=1;

 rs=a;
         re=a+(bs-1);
         cs=b;
         ce=b+(bs-1);


 sum=0;
       for m=rs:re %3

           co=0;


           for n=cs:ce  %4
               co=co+1;



               bb(ro,co)=im(m,n) ;
%                sum=sum+bb(ro,co);
```

```
     iy=iy+1;

        end %4
            ro=ro+1 ;

        end%3

%sum=sum/(bs*bs);
as=mean(mean(abs(bb)));
            im(a+1,b+1)=as;      %eq(9)
%     ma(de)=max(max( as));
            de=de+1;
    end
end
%/////
%directional feature
%equation 12
imm=im;
  wq=max(imm);

imm1=abs(log(imm));


%/////
%finding total energy  Equation(14)
%s=size(Act);
bs=9%input('Enter block size')

%imm=;
e1=floor(ss(1,1)/bs);
e2=floor(ss(1,2)/bs);
de=1;
for a=1:e1   %1

    for b=1:e2  %2
iy=1;
 ro=1;
 co=1;

         rs=bs*a-(bs-1);
         re=bs*a;
         cs=bs*b-(bs-1);
         ce=bs*b;

         summ=0;
        for m=rs:re %3

            co=0;
            for n=cs:ce  %4
                co=co+1;
```

```
                bb(ro,co)=imm(m,n) ;
                summ=summ+bb(ro,co);
     iy=iy+1;

        end %4
            ro=ro+1 ;

        end%3


sa(a,b)=summ./81;

%sum=sum/(bs*bs);
%as=mean(mean(abs(bb)));
 %           im(a+1,b+1)=as;      %eq(9)
%     ma(de)=max(max( as));
            de=de+1;
    end
end
%/////

%ETot2=sa;

%percentage energy
bs=9%input('Enter block size')

%imm=nma;
e1=floor(ss(1,1)/bs);
e2=floor(ss(1,2)/bs);
de=1;
for a=1:e1   %1

    for b=1:e2  %2
iy=1;
 ro=1;
 co=1;


         rs=bs*a-(bs-1);
         re=bs*a;
         cs=bs*b-(bs-1);
         ce=bs*b;


         sum=0;
        for m=rs:re %3

            co=0;
            for n=cs:ce  %4
```

```
                co=co+1;
                pimm1(m,n)=imm(m,n)/sa(a,b) ;
     iy=iy+1;

        end %4
            ro=ro+1 ;

        end%3
            de=de+1;
    end
end
%/////
%weighted roughness
  wr090=wq*pimm1;


  it=1;
for x=1:ss(1,1)
for y=1:ss(1,2)
    w45(x,y)=(-(x/s)*exp(-(x*x+y*y)/(2*s*s)))*cos(45)+(-(y/s)*exp(-
(x*x+y*y)/(2*s*s)))*sin(45);
   %c(1,it)=w0(x,y);
   it=it+1;
   end
end
it=1;
fw45=conv2(w45,I2,'full');
fw45=fw45(1:ss(1,1),1:ss(1,2));
fw45=fw45;
bs=9;
nma=fw45;
im=abs(fw45);

e1=ss(1,1)-(bs-1);
e2=ss(1,2)-(bs-1);

de=1;
for a=1:e1   %1

    for b=1:e2  %2


iy=1;
 ro=1;
 co=1;

 rs=a;
         re=a+(bs-1);
         cs=b;
         ce=b+(bs-1);
```

```
 sum=0;
        for m=rs:re %3

            co=0;


            for n=cs:ce  %4
                co=co+1;



                bb(ro,co)=im(m,n) ;
%                sum=sum+bb(ro,co);

     iy=iy+1;

        end %4
            ro=ro+1 ;

        end%3

%sum=sum/(bs*bs);
as=mean(mean(abs(bb)));
            im(a+1,b+1)=as;      %eq(9)
%     ma(de)=max(max( as));
            de=de+1;
    end
end
%/////
%directional feature
%equation 12
imm=im;
  wq=max(imm);

imm1=abs(log(imm));


%/////
%finding total energy  Equation(14)
%s=size(Act);
bs=9%input('Enter block size')

%imm=;
e1=floor(ss(1,1)/bs);
e2=floor(ss(1,2)/bs);
de=1;
for a=1:e1   %1
```

```
    for b=1:e2  %2
iy=1;
 ro=1;
 co=1;

          rs=bs*a-(bs-1);
          re=bs*a;
          cs=bs*b-(bs-1);
          ce=bs*b;

          summ=0;
         for m=rs:re %3

             co=0;
             for n=cs:ce  %4
                 co=co+1;
                 bb(ro,co)=imm(m,n) ;
                 summ=summ+bb(ro,co);
     iy=iy+1;

         end %4
             ro=ro+1 ;

         end%3


sa(a,b)=summ./81;

%sum=sum/(bs*bs);
%as=mean(mean(abs(bb)));
 %           im(a+1,b+1)=as;     %eq(9)
%     ma(de)=max(max( as));
             de=de+1;
    end
end
%/////

%ETot2=sa;

%percentage energy
bs=9%input('Enter block size')

%imm=nma;
e1=floor(ss(1,1)/bs);
e2=floor(ss(1,2)/bs);
de=1;
for a=1:e1   %1
```

```
    for b=1:e2  %2
iy=1;
 ro=1;
 co=1;


          rs=bs*a-(bs-1);
          re=bs*a;
          cs=bs*b-(bs-1);
          ce=bs*b;


          sum=0;
         for m=rs:re %3

             co=0;
             for n=cs:ce  %4
                 co=co+1;
                 pimm2(m,n)=imm(m,n)/sa(a,b) ;
     iy=iy+1;

         end %4
             ro=ro+1 ;

         end%3
             de=de+1;
    end
end
%/////
%weighted roughness
  wr45=wq*pimm2;



  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%OOOOOOOOOOOOOOOOOOOO



pimm=(pimm);%+pimm1+pimm2)/3;

  %!!!!!

s1=.8;
s2=1;
s3=1.2;
s4=1.5;
s=size(I2);
```

```
fgo=1;
for gt=1:size(pimm,1)
    for gt1=1:size(pimm,2)

    Amean(fgo)=pimm(gt,gt1);
     fgo=fgo+1;
    end
end



%   classefgo=1;
for gt=1:s(1,1)
    for gt1=1:s(1,2)
        neimean(fgo)=pimm(gt,gt1);
        fgo=fgo+1;
    end
end



beta=1;
sa=size(neimean);
for gt=1:sa(1,2)
    if neimean(gt)*beta ==0
        delmean(gt)=1;
    else
        delmean(gt)=0;
    end

end

%   creating classes
for i=1:s(1,1)*s(1,2)  %1

        if Amean(i)<=s1
            cl(i)='A'; %k1
            cpp(i)=0;
        end

        if Amean(i)>s1 & Amean(i)<=s2
            cl(i)='B'; %k1
            cpp(i)=100;
        end

        if Amean(i)>s2 & Amean(i)<=s3
            cl(i)='C'; %k1
            cpp(i)=150;
        end

        if Amean(i)>s3 & Amean(i)<=s4
            cl(i)='D'; %k1
```

```
            cpp(i)=200;
        end

        if Amean(i)>s4
            cl(i)='E'; %k1
            cpp(i)=255;
        end


end %1
vis=1;

for i=1:s(1,1)
    for jj=1:s(1,2)
        vd(i,jj)=cl(vis);
        vdr(i,jj)=cpp(vis);
        vis=vis+1;
    end
end



Ac=cl;

% mean for all pixel
sum=0;
for i=1:s(1,1)*s(1,2) %1
    if Ac(i)=='A'
        sum=sum+Amean(i);
    end
end %1
MeanA=sum/(s(1,1)*s(1,2));

sum=0;
for i=1:s(1,1)*s(1,2)%2
    if Ac(i)=='B'
        sum=sum+Amean(i);
    end
end %2
MeanB=sum/(s(1,1)*s(1,2));

sum=0;
for i=1:s(1,1)*s(1,2)%3
    if Ac(i)=='C'
        sum=sum+Amean(i);
    end
end%3
MeanC=sum/(s(1,1)*s(1,2));

sum=0;
```

```
for i=1:s(1,1)*s(1,2) %2
    if Ac(i)=='D'
        sum=sum+Amean(i);
    end
end %2
MeanD=sum/(s(1,1)*s(1,2));

sum=0;
for i=1:s(1,1)*s(1,2)%1
    if Ac(i)=='E'
        sum=sum+Amean(i);
    end
end%1
MeanE=sum/(s(1,1)*s(1,2));

%/////////

%var values

sum=0;
for i=1:s(1,1)*s(1,2)%1
    if Ac(i)=='A'
        sum=sum+(Amean(i)-MeanA)^2;
    end
end%1
varA=sum/s(1,1)*s(1,2);

sum=0;
for i=1:s(1,1)*s(1,2) %2
    if Ac(i)=='B'
        sum=sum+(Amean(i)-MeanB)^2;
    end
end %2
varB=sum/s(1,1)*s(1,2);

sum=0;
for i=1:s(1,1)*s(1,2)%3
    if Ac(i)=='C'
        sum=sum+(Amean(i)-MeanC)^2;
    end
end%3
varC=sum/s(1,1)*s(1,2);

sum=0;
for i=1:s(1,1)*s(1,2)%4
    if Ac(i)=='D'
        sum=sum+(Amean(i)-MeanD)^2;
    end
end%4
varD=sum/s(1,1)*s(1,2);
```

```
sum=0;
for i=1:s(1,1)*s(1,2)%5
    if Ac(i)=='E'
        sum=sum+(Amean(i)-MeanE)^2;
    end
end%5
varE=sum/s(1,1)*s(1,2);

%normalization factor


tot=s(1,1)*s(1,2);

nof=1;
as=(2*3.14)^nof;
Z(1)=sqrt(as*varA);
Z(2)=sqrt(as*varB);
Z(3)=sqrt(as*varC);
Z(4)=sqrt(as*varD);
Z(5)=sqrt(as*varE);
%fdgddfdfgf
%energy function
sum1=0;
sum2=0;
sum3=0;
sum4=0;
sum5=0;
c1=1;
c2=1;
c3=1;
c4=1;
c5=1;
for i=1: tot %1

        if Amean(i)<=s1

            sum1=sum1+(Amean(i)-MeanA)^2;
%            cl(i)='A'; %k1

            c1=c1+1;


        end

        if Amean(i)>s1 & Amean(i)<=s2
            sum2=sum2+(Amean(i)-MeanB)^2;
            c2=c2+1;
            %cl(i)='B'; %k1
        end
```

```
            if Amean(i)>s2 & Amean(i)<=s3
             sum3=sum3+(Amean(i)-MeanC)^2;
             c3=c3+1;
           %  cl(i)='C'; %k1
        end

        if Amean(i)>s3 & Amean(i)<=s4
             sum4=sum4+(Amean(i)-MeanD)^2;
             c4=c4+1;
        %    cl(i)='D'; %k1
        end

        if Amean(i)>s4
             sum5=sum5+(Amean(i)-MeanE)^2;
             c5=c5+1;
           % cl(i)='E'; %k1
        end
end %1


E(1)=sum1/c1;
E(2)=sum2/c2;
E(3)=sum3/c3;
E(4)=sum4/c4;
E(5)=sum5/c5;

%conditional probability
for i=1:tot%1

        if Amean(i)<=s1
           % po=(Amean(i)-MeanA)^2
           %po1=po/(2*varA);
           po1=E(1)/(2*varA);
            po2=log(as*varA);
            po3=-(po1-.5*po2)-delmean(i);%+neimean1(i);
%            po3=-(po1-.5*po2);%+neimean1(i);
            pcl(i)= exp(po3) ; %k1
         %  pcl(i)=po3;
        end

        if Amean(i)>s1 & Amean(i)<=s2
            %po=(Amean(i)-MeanB)^2
            %po1=po/(2*varB);
             po1=E(2)/(2*varB);
            po2=log(as*varB);
            po3=-(po1-.5*po2)-delmean(i);%+neimean1(i);
%            po3=-(po1-.5*po2);%+neimean1(i);
            pcl(i)= exp(po3) ; %k
           %  pcl(i)=po3;
     end
        if Amean(i)>s2 & Amean(i)<=s3
```

```
            %po=(Amean(i)-MeanC)^2
            %po1=po/(2*varC);
             po1=E(3)/(2*varC);
            po2=log(as*varC);
            po3=-(po1-.5*po2)-delmean(i);%+neimean1(i);
  %                      po3=-(po1-.5*po2);%+neimean1(i);
            pcl(i)= exp(po3) ; %
         % pcl(i)=po3;
        end

        if Amean(i)>s3 & Amean(i)<=s4
 %            po=(Amean(i)-MeanD)^2
%            po1=po/(2*varD);
                po1=E(4)/(2*varD);
            po2=log(as*varD);
            po3=-(po1-.5*po2)-delmean(i);%+neimean1(i);
 %           po3=-(po1-.5*po2);%+neimean1(i);
            pcl(i)= exp(po3) ; %
%            pcl(i)=po3;
        end
         if Amean(i)>s4
%            po=(Amean(i)-MeanE)^2
 %           po1=po/(2*varE);
             po1=E(5)/(2*varE);
            po2=log(as*varE);
            po3=-(po1-.5*po2)-delmean(i);%+neimean1(i);
%                       po3=-(po1-.5*po2)+neimean1(i);
            pcl(i)= exp(po3) ; %
         %    pcl(i)=po3;
        end
end %1


%fgdfgdfgdfg
%conditional probability
disp('dkfjdjf')
vis=1;

for i=1:s(1,1)
    for jj=1:s(1,2)
        vdd(i,jj)=uint8(pcl(vis)+100);
        vis=vis+1;
    end
end
%figure
imshow(vdd)
imwrite(vdd,'KA.SU3.38_output_gabour.jpg')
%////////
save pimm18.mat pcl
```