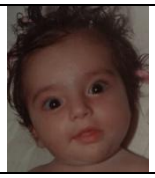
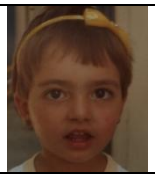
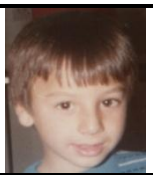
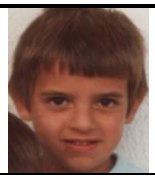
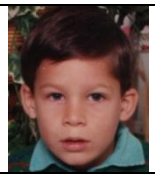
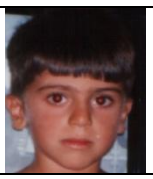
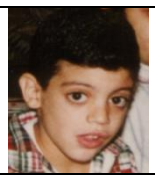
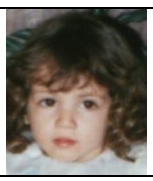
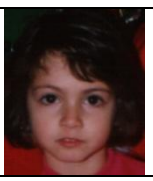
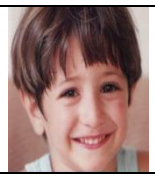
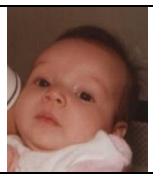
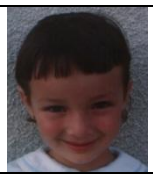
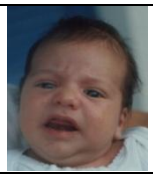
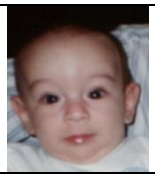
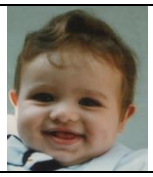
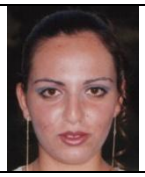
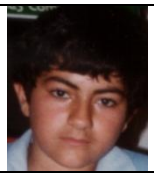
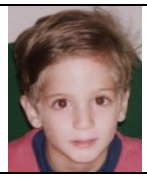
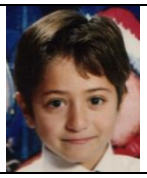
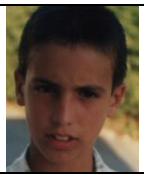
Lampiran A: Data Citra

A.1 Citra Data Latih

					
009A00	009A03	009A05	015A00	015A03	015A05
					
019A00	019A03	019A05	023A00	023A03	023A05
					
035A00	035A03	035A07	036A00	036A03	036A05
					
038A00	038A03	038A05	040A00	040A03	040A05
					
043A00	043A03	043A05	058A00	058A03	058A05
					
069A00	069A03	069A05	070A00	070A03	070A05
					
073A00	073A03	073A05	074A00	074A03	074A05

					
075A00	075A03	075A05	076A00	076A03	076A05
					
078A00	078A03	078A05	079A00	079A03	079A05
					
080A00	080A03	080A05	081A00	080A03	080A05

A.2 Citra Data Uji

					
009A18	015A19	019A07	023A25	035A18	036A19
					
038A10	040A09	043A15	058A13	069A07	070A09
					
073A06	074A06	075A07	076A10	078A11	079A07
					
080A14	081A07				

Lampiran B: List Program

Simulasi

```
function varargout = simulasi(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @simulasi_OpeningFcn, ...
                   'gui_OutputFcn',  @simulasi_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
function simulasi_OpeningFcn(hObject, eventdata, handles, varargin)

handles.output = hObject;

guidata(hObject, handles);

function varargout = simulasi_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;

function edit5_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function pushbutton4_Callback(hObject, eventdata, handles)
run mainMenu;
close ('simulasi');

function pushbuttonEkstraksiCiri_Callback(hObject, eventdata, handles)
    clc;
    global open_image imagepath hasil point;
    imagename = open_image;
    imagename = strrep(imagename, '.jpg', '');
    imagename = strrep(imagename, '.bmp', '');
    imagename = strrep(imagename, '.JPG', '');
    imagename = strrep(imagename, '.BMP', '');

    [hasil point] = ekstraksi_point(strcat(imagepath,imagename));
    axes(handles.axesFeaturePoint);
    imshow(hasil);

    image=imread([imagepath imagename '_normal.jpg']);
```

```
    image=uint8(image);
    % b=rgb2gray(a);
    hasilGabor = two_d_gaborwavelet(image);
    hasilGabor = hasilGabor';
    xlswrite([imagepath imagename '_gabor.xls'], hasilGabor);
    set(handles.edit26,'string', hasilGabor(1));
    set(handles.edit27,'string', hasilGabor(2));
    set(handles.edit28,'string', hasilGabor(3));
    set(handles.edit29,'string', hasilGabor(4));
    set(handles.edit30,'string', hasilGabor(5));
    set(handles.edit31,'string', hasilGabor(6));
    set(handles.edit32,'string', hasilGabor(7));
    set(handles.edit33,'string', hasilGabor(8));
    set(handles.edit34,'string', hasilGabor(9));
    set(handles.edit35,'string', hasilGabor(10));
    set(handles.edit36,'string', hasilGabor(11));
    set(handles.edit37,'string', hasilGabor(12));
    set(handles.edit38,'string', hasilGabor(13));
    set(handles.edit39,'string', hasilGabor(14));
    set(handles.edit40,'string', hasilGabor(15));

    gabor = hasilGabor;
    point = (point/100);

    hasil_ekstraksi = point + gabor;
    hasil_ekstraksi = hasil_ekstraksi/2;
    xlswrite([imagepath imagename '_ekstraksi.xls'], hasil_ekstraksi);
    msgbox('Proses Ekstraksi Ciri telah berhasil dilaksanakan','', 'none');
function edit8_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function PathUji_Sim_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function pushbuttonBrowse_Callback(hObject, eventdata, handles)

    clc;
    global open_image imagepath;
    [open_image,imagepath]=uigetfile(...
        {'*.bmp;*.jpg;','File Citra (*.bmp,*jpg,)';
        '*.bmp','File Bitmap (*.bmp)';...
        '*.jpg','File jpeg (*.jpg)';
        '*.*','Semua File (*.*)'},'Pick an Image');

    clc;
    disp_image='false';
    fopen=strcat(imagepath,open_image);
```

```
    handles.fopen=fopen;
    guidata(hObject,handles);
    if ~isequal(open_image,0)
        global X;
        X=imread(fopen);
        XInfo=imfinfo(fopen);
        disp_image='true';
    end

if isequal(disp_image,'true')
    % displaying image ----------------------------------------------
    handles.banner = X;
    axes(handles.axesOriginalImage);
    imshow(handles.banner);
    file= [imagepath, open_image];
    set(handles.axesOriginalImage, 'Visible', 'off', 'Units', 'pixels');
    set(handles.NamaDataUji, 'Visible', 'on', 'string', open_image);
    set(handles.PathUji_Sim,'string', file);
    % save parameter  ----------------------------------------------
    handles.oriIm2=X;
    oriIm2=X;
    X=double(X);
    oriIm   =X;
    oriName =open_image;
    proc='encode';
    handles.oriIm=oriIm;
    handles.oriName=oriName;
    handles.proc=proc;
    guidata(hObject,handles);
    tes = fopen(1:end-4);
end
function edit26_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes during object creation, after setting all properties.
function edit27_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes during object creation, after setting all properties.
function edit28_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes during object creation, after setting all properties.
function edit29_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end

% --- Executes during object creation, after setting all properties.
function edit30_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes during object creation, after setting all properties.
function edit31_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes during object creation, after setting all properties.
function edit32_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes during object creation, after setting all properties.
function edit33_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes during object creation, after setting all properties.
function edit34_CreateFcn(hObject, eventdata, handles)
.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes during object creation, after setting all properties.
function edit35_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes during object creation, after setting all properties.
function edit36_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes during object creation, after setting all properties.
function edit37_CreateFcn(hObject, eventdata, handles)
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes during object creation, after setting all properties.
function edit38_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes during object creation, after setting all properties.
function edit39_CreateFcn(hObject, eventdata, handles)
), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes during object creation, after setting all properties.
function edit40_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in pushbutton.
function pushbuttonPreprocessing_Callback(hObject, eventdata, handles)
    global X open_image imagepath;
    imagename = open_image;
    imagename = strrep(imagename, '.jpg', '');
    imagename = strrep(imagename, '.bmp', '');
    imagename = strrep(imagename, '.JPG', '');
    imagename = strrep(imagename, '.BMP', '');
    resultname = 'preprocess.jpg';
    X = uint8(X);
    if (~isgray(X))
        X = rgb2gray(X);
    end


    threshold = 10;

    s = fdmex(X', threshold);

    sorteds = sortrows(s,3);
    sizes = size(sorteds,1);
    if sizes>0
        start1 = floor(sorteds(sizes,1)-sorteds(sizes,3)/2);
        start2 = floor(sorteds(sizes,2)-sorteds(sizes,3)/2);
        psize  = sorteds(sizes,3);
        imgtemp = X(start2:start2+psize, start1:start1+psize);
        imwrite(imgtemp,[imagepath imagename '_normal.jpg'],'jpg');
        axes(handles.axesNormalImage);
        imshow(imgtemp);
```

```
%         msgbox('Proses Normalisasi telah berhasil dilaksanakan','',
'none');
    end


% --- Executes on button press in pushbuttonIdentifikasi.
function pushbuttonIdentifikasi_Callback(hObject, eventdata, handles)
global open_image imagepath;
imagename = open_image;
imagename = strrep(imagename, '.jpg', '');
imagename = strrep(imagename, '.bmp', '');
imagename = strrep(imagename, '.JPG', '');
imagename = strrep(imagename, '.BMP', '');

%testing JST%
load 'hasil_training';
P = xlsread([imagepath imagename '_ekstraksi.xls'], 'Sheet1');
% P=xlsread('hasil ekstraksi_uji_energi_17x17.xls','Sheet2')
[output,Pf,Af,e,perf]=sim(net,P,[],[])

% simpan hasil testing JST
fileSAVE = [imagepath imagename '.mat' ];
save(fileSAVE, 'output', 'e', 'perf');

[sortArray idx] = sort(output, 'descend');
hasilValue = sortArray(1)
hasilIndex = idx(1)

load('subjects.mat');
subjectName = subjects(hasilIndex,1);
subjectImg1 = subjects(hasilIndex,2);
subjectImg2 = subjects(hasilIndex,3);
subjectImg3 = subjects(hasilIndex,4);

axes(handles.axes10);
img = imread(char(subjectImg1));
imshow(img);

axes(handles.axes18);
img = imread(char(subjectImg2));
imshow(img);

axes(handles.axes19);
img = imread(char(subjectImg3));
imshow(img);

set(handles.edit41,'string', hasilIndex);
set(handles.edit45,'string', char(subjectName));



% --- Executes during object creation, after setting all properties.
function edit41_CreateFcn(hObject, eventdata, handles)
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton16.
function pushbutton16_Callback(hObject, eventdata, handles)
global hasil;
figure;
imshow(hasil);



function edit45_Callback(hObject, eventdata, handles)
function edit45_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton27.
function pushbutton27_Callback(hObject, eventdata, handles)

global point;

tabel_ekstraksi_point(point);


% --- Executes on button press in pushbutton28.
function pushbutton28_Callback(hObject, eventdata, handles)
x = imread('white.bmp');
axes(handles.axesOriginalImage);
imshow(x);
axes(handles.axesNormalImage);
imshow(x);
axes(handles.axesFeaturePoint);
imshow(x);
axes(handles.axes10);
imshow(x);
axes(handles.axes18);
imshow(x);
axes(handles.axes19);
imshow(x);

set(handles.edit41,'string', '');
set(handles.edit45,'string', '');
set(handles.edit26,'string', '');
set(handles.edit27,'string', '');
set(handles.edit28,'string', '');
set(handles.edit29,'string', '');
set(handles.edit30,'string', '');
set(handles.edit31,'string', '');
```

```
set(handles.edit32,'string', '');
set(handles.edit33,'string', '');
set(handles.edit34,'string', '');
set(handles.edit35,'string', '');
set(handles.edit36,'string', '');
set(handles.edit37,'string', '');
set(handles.edit38,'string', '');
set(handles.edit39,'string', '');
set(handles.edit40,'string', '');
set(handles.PathUji_Sim,'string', '');
set(handles.NamaDataUji,'string', '');
```

2D Gabor wavelet

```
function [template]=two_d_gaborwavelet(image)

alpha=1/sqrt(2);
i=1;
for teta=pi/6:pi/6:pi/2
    for j=2:6

        for x=-5:5
            for y=-5:5
                h=(1/(2*pi))*exp((-(alpha^(2*j))*(x^2+y^2)/2));
                h1(x+9,y+9)=h;

                R=cos(pi*(alpha^j)*(x*cos(teta)+y*sin(teta)));
                R1(x+9,y+9)=R;
                size(R1);
                I=sin(pi*(alpha^j)*(x*cos(teta)+y*sin(teta)));
I1(x+9,y+9)=I;
            end
        end
         Rh=h1.*R1;
         Ih=h1.*I1;
         Ro=conv2(image,Rh,'same');

         Io=conv2(image,Ih,'same');
   out=sqrt(Ro.*Ro+Io.*Io);

   out_normalisasi= abs(out)/(max(max(out)));

      [m n]=size(out_normalisasi);
   resolusi=m*n;
   power=sum(sum(out_normalisasi.^2));
   energi=power/resolusi;
   template(i)= energi;
   i=i+1;
    end
end
```

Training JST

```
function training_jst_bp;
P=xlsread('hasil ekstraksi_latih.xlsx','Sheet1')

% load kelas target yang digunakan
load 'kelas_target_latih_20_neuron';
t=target_latih_20_neuron;

% pembentukan arsitrektur JST
net=newff(minmax(P),[150,200,20],{'tansig','logsig','purelin'},'traincgf');

% inisialisasi parameter-parameter training JST
net.trainParam.epochs=2000;          % jumlah eppoch yang diharapkan
net.trainParam.goal=1E-6;            % mse yang diharapkan
net.trainParam.lr=0.8;               % lr yang digunakan
net.trainParam.show =20;             % setiap kelipatan 10 eppoch,
ditampilkan hasil training

% training JST
[net,tr,Y,E,Pf,Af]=train(net,P,t,[],[]);

% simpan hasil training
save hasil_training_1 net;
```

Tinytest

```
clc;
imageFolder = 'image data latih';
resultFolder = 'data_latih normal';
[path name] = DirBrowser(imageFolder);

for wew=1:length(path)
    imname = char(path(wew));
    x = imread(imname);
    if (~isgray(x))
        x = rgb2gray(x);
    end

    % decision threshold.
    % change this to a smaller value, if too many false detections occur.
    % change it to a larger value, if faces are not recognized.
    % a reasonable range is -10 ... 10.
    threshold = 0;

    %figure;
    hold on;
    colormap gray;
    s = fdmex(x', threshold);

    sorteds = sortrows(s,3);
    sizes = size(sorteds,1);
    if sizes>0
        start1 = floor(sorteds(sizes,1)-sorteds(sizes,3)/2);
        start2 = floor(sorteds(sizes,2)-sorteds(sizes,3)/2);
```

```
        psize  = sorteds(sizes,3);
        imgtemp = x(start2:start2+psize, start1:start1+psize);
        imwrite(imgtemp,[resultFolder '\' char(name(wew))],'jpg');
    end
    axis equal;
    axis off
end
```

Testing JST

```
function testing_jst_bp;
% deskripsi     : melakukan testing atau testing JST terhadap data testing.
% input         : data testing, data hasil training JST, data target.
% proses        : data testing masuk ke dalam JST yang telah dtraining
sebelumnya kemudian dengan pemrosesan
%                 yang bersifat "black box" dihasilkan output. Dari nilai
%                 output tersebut dapat ditentukan data testing tersebut
masuk
%                 ke dalam kelas target yang mana.
% output        : satu nilai yang kemudian digunakan untuk mengetahui
termasuk kelas target mana data testing tersebut.
% ========================================================================

% load data yang digunakan
% load 'jumlah_data.mat';
% load 'data_testing.mat';
load 'hasil_training_1';
% load 'kelas_target_testing';
% load data yang digunakan
% load 'kelas_target_uji_35_neuron';

P=xlsread('hasil ekstraksi_uji_energi.xls','Sheet1')
% P=abs(PP)/(max(max(PP)))
% tentukan jumlah data testing
% jumlah_vektor_input=n_data_testing;

% load data testing
   % inisiaisasi awal matrik A1
%      A1=[];
%
%    % tentukan data training A1 yang diload dari data_testing.mat
%      for   i= 1:jumlah_vektor_input
%            A1=[A1;data_testing{i}];             %ditranspose baris
%      end
%      A=A1';
%      size(A)

% load kelas target testing
   % inisiaisasi awal matrik t1
%      A=target_uji_35_neuron;

  % tentukan data training t yang diload dari kelas_target_training.mat
%     U=cell2mat(kelas_target_testing);
%     size(U)
%
%testing JST
```

```
[output,Pf,Af,e,perf]=sim(net,P,[],[])

% simpan hasil testing JST
save hasil_testing output e perf;
```

Feature Point

```
function [hasil result] = ekstraksi(imagename)
    fileName = imagename;
    loadData = importdata([fileName '.pts'],' ');
    result = zeros(15,1);

    result(1) = sqrt((loadData(28,1)-loadData(30,1))^2 + (loadData(28,2)-
loadData(30,2))^2);
    result(2) = sqrt((loadData(35,1)-loadData(33,1))^2 + (loadData(35,2)-
loadData(33,2))^2);
    result(3) = sqrt((loadData(28,1)-loadData(33,1))^2 + (loadData(28,2)-
loadData(33,2))^2);
    result(4) = sqrt((loadData(30,1)-loadData(35,1))^2 + (loadData(30,2)-
loadData(35,2))^2);
    result(5) = sqrt((loadData(28,1)-loadData(68,1))^2 + (loadData(28,2)-
loadData(68,2))^2);
    result(6) = sqrt((loadData(32,1)-loadData(68,1))^2 + (loadData(32,2)-
loadData(68,2))^2);
    result(7) = sqrt((loadData(30,1)-loadData(68,1))^2 + (loadData(30,2)-
loadData(68,2))^2);
    result(8) = sqrt((loadData(35,1)-loadData(68,1))^2 + (loadData(35,2)-
loadData(68,2))^2);
    result(9) = sqrt((loadData(37,1)-loadData(68,1))^2 + (loadData(37,2)-
loadData(68,2))^2);
    result(10) = sqrt((loadData(33,1)-loadData(68,1))^2 + (loadData(33,2)-
loadData(68,2))^2);
    result(11) = sqrt((loadData(40,1)-loadData(68,1))^2 + (loadData(40,2)-
loadData(68,2))^2);
    result(12) = sqrt((loadData(44,1)-loadData(68,1))^2 + (loadData(44,2)-
loadData(68,2))^2);
    result(13) = sqrt((loadData(49,1)-loadData(68,1))^2 + (loadData(49,2)-
loadData(68,2))^2);
    result(14) = sqrt((loadData(52,1)-loadData(68,1))^2 + (loadData(52,2)-
loadData(68,2))^2);
    result(15) = sqrt((loadData(55,1)-loadData(68,1))^2 + (loadData(55,2)-
loadData(68,2))^2);



    xlswrite([fileName '_point.xls'], result);


    image = imread([fileName '.jpg']);
    if (~isgray(image))
        image = rgb2gray(image);
    end

    temp = zeros(size(image,2),size(image,1));
    for (i=1:length(loadData))
        temp(floor(loadData(i,1)),floor(loadData(i,2))) = 255;
    end
```

```
    temp = rot90(temp,3);
    temp = flipdim(temp,2);

    hasil = double(image) + temp;
    hasil = uint8(hasil);
end
```