LAMPIRAN A

SOURCE CODE

```
clear;
close all;
clc;

sir=[0 3 1 0 3 0 2 1 3 2 1 0 2 1 3 1 0 2 1 0 2 3 2 3 0
2 1 0 3 0 2 1 3 1 0 2 1 0 2 1 0 2 3 1 0 3 0 2 1 0 2 1 0
2 1 0 3 0 2 1 0 2 1 0 2 1 0 3 2 3 1 0 2 1 0 3 1 0 2 3 1
3 2 1 0 2 3 0 2 1 0 2 1 0 2 1 0 2 3 3 0]
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 state model pertama
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=0:3
    histo(i+1)=sum(sir==i);
end
histo
pist=histo/sum(histo);
h1(1)=-sum(pist.*log2(pist))

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
state model ke dua
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
warning off
A=[0 0 1 0
    3 0 3 0 ];
hh2=[];Aaa=[];
for machines=0:(2^8-1)
    aa= bitget(machines,8:-1:1);
    A = [aa(1:4); aa(5:8)];
    s_crt=0;
    for i=1:length(sir)
        symbol=sir(i);
        state(i)=s_crt;
```

```
        %s_next=mod1(s_crt,symbol,A);
        s_next=A(s_crt+1,symbol+1);
        s_crt=s_next;
    end
    for i=0:1
        for j=0:3
            counti(i+1,j+1)=sum((state==i).*(sir==j));
        end
    end
    for i=0:1
        for j=0:3
            counti(i+1,j+1)=sum((state==i).*(sir==j));
        end
        histo=counti(i+1,:);
        ivalid=find(histo>0);
        pist=histo/sum(histo);
        h2(i+1)=-sum(pist(ivalid).*log2(pist(ivalid)));
    end
    pstate=sum(counti')/sum(sum(counti));
    hh2=[hh2 sum(pstate.*h2)];
    hh3(machines+1)=sum(pstate.*h2);
    Aaa=[Aaa;A];
end

[val ind]=sort(hh3);
for i=1:length(ind)
    figure(i)
    machines=ind(i)-1;
    aa= bitget(machines,8:-1:1);
    A = [aa(1:4); aa(5:8)];
    s_crt=0;
    for i=1:length(sir)
```

```
        symbol=sir(i);
        state(i)=s_crt;
        %s_next=mod1(s_crt,symbol,A);
        s_next=A(s_crt+1,symbol+1);
        s_crt=s_next;

        if(s_next==3)
            error3
        end
    end
    for i=0:1
        for j=0:3
            counti(i+1,j+1)=sum((state==i).*(sir==j));
        end
    end
    for i=0:1
        for j=0:3
            counti(i+1,j+1)=sum((state==i).*(sir==j));
        end
        histo=counti(i+1,:);
        ivalid=find(histo>0);
        pist=histo/sum(histo);
        h2(i+1)=-sum(pist(ivalid).*log2(pist(ivalid)));
    end
    pstate=sum(counti')/sum(sum(counti));
    hh2=[hh2 sum(pstate.*h2)];
%clf
    %hh=plot(0,5,'o','MarkerSize',20)
    xx=1:500;
    circ=[];elip=[];
    circ(1,:)= 5*cos(2*pi*xx/500);
    circ(2,:)= 5*sin(2*pi*xx/500);
```

```
 elip(1,:)= 8*cos(2*pi*xx/520-pi/2+pi/15);
 elip(2,:)= 20*sin(2*pi*xx/520-pi/2+pi/15);
 elip=[elip(1,:) elip(1,1)+3*(0:0.1:1)
 elip(1,1)+1*(0:0.1:1)
 elip(2,:) elip(2,1)+1*(0:0.1:1)
 elip(2,1)+3*(0:0.1:1)];

plot(0,-30,'.k',100,70,'k.')
hold on
hhp=plot(circ(1,:), 5+circ(2,:),'-b','LineWidth',3)
if(sum(A(1,:)==0)>0)
    plot(elip(1,:), 30+elip(2,:))
end

plot(60+circ(1,:), 5+circ(2,:),'-b','LineWidth',3)
if(sum(A(2,:)==1)>0)
    plot(60+elip(1,:), 30+elip(2,:),'-b')
end
if(sum(A(1,:)==1)>0)
    plot( [60-9 9],[8 8],'b-',[60-9-3 60-9],
      [8+1.5 8],'b-',[60-9-3 60-9],[8-1.5 8],'b-')
end
if(sum(A(2,:)==0)>0)
    plot( [60-9 9],[2 2],'b-',[9+3 9],[2+1.5 2],
           'b-',[9+3 9],[2-1.5 2],'b-')
end
for i=1:4
    if( A(1,i)==0 )
        hhpp=text(11,46-(i-1)*4,[6*16+i   '   n_'
6*16+i '=' num2str(counti(1,i))],'FontSize',14)
    end
end
```

```
    for i=1:4
        if(A(1,i)==1)
            text(21,25-(i-1)*4,[6*16+i   '   n_' 6*16+i
'=' num2str(counti(1,i))],'FontSize',14)
        end
    end
    for i=1:4
        if(A(2,i)==0)
            text(21,-5-(i-1)*4,[6*16+i   '   n_' 6*16+i
'=' num2str(counti(2,i))],'FontSize',14)
        end
    end
    for i=1:4
        if(A(2,i)==1)
            text(72,46-(i-1)*4,[6*16+i   '   n_' 6*16+i
'=' num2str(counti(2,i))],'FontSize',14)
        end
    end
    text(60,-20,['Entropy = '
num2str(sum(pstate.*h2))],'FontSize',14)
 A
pause
 axis off
end
```

```
function [x1pos, x2pos] = bivalue2(input)

% masukan matrix ke 1D array
[SR, SC] = size(input);
input = reshape(input, 1, SR*SC);

% N adlah histogram dari suatu gambar
%gambar(200);
edge =linspace(0,255,256);
N = histc(input,edge);
%bar(edge,N,'histc');

count=0; %counter untuk merekam histogram
temp1 =1; %memsukan ke1 bivalue
temp2 =1;  %memsukan ke2 bivalue

peaks = [N(1)];
peakpos = [1];
for i=2:1:255
%   if(N(i) == 0)
%      count = count+1;
%   end
    if (N(i)>N(i+1)) & (N(i)>N(i-1))
      peaks=[peaks N(i)];
    peakpos = [peakpos i];
   end

end

peaks = [peaks N(256)];
peakpos = [peakpos 256];

x1 = max(peaks);
r = find(peaks == x1);
x1pos = peakpos(r(1))-1;
%if length(r)>1
%   x2 = x1;
%%   x2pos = x1pos;
 %  return;
%end
peaks=[peaks(1:r-1) peaks(r+1:length(peaks))];
x2= max(peaks);
r = find(peaks == x2);
x2pos = peakpos(r(1))-1;
```

```
function [class, mu, sigma] = classify64(I)
%CLASSIFY64 kelas pertama postal image blocks.
%   [CLASS,MU,SIGMA] = CLASSIFY64(I) performa pertam
classification dari I
%   katagori: gambar, text, background, or
undetermined.
%
%
%   1 = gambar
%   2 = text
%   3 = background
%   4 = undetermined
%
%   MU arti dari  block and SIGMA adalah standar
deviasi.
%
%   I adalah matrix dari piksel  grayscale gambar
dengan ukuran 64x64.
%
%   selalu melihat POSTALSEG, FIRSTPASS, FEATURES

A = features(I);

% background
if A.sigma<10
    class = 3;
% text
elseif (A.Dx>25 & A.Dy>25 & A.alpha>300) | (A.Dx>50 &
A.Dy>50)
    class = 2;
% gambar
elseif A.sigma>25 & (A.alpha<200 | A.gc<1e4)
    class = 1;
% undetermined
else
    class = 4;
end

mu=A.mu;
sigma=A.sigma;
```

```
function [Dx,Dy] = DxDy(I)
%DXDY kalkulasi dai penjumlahan magnituda dari postal
gambar.
%   [DX,DY] = DXDY(I) perkalian magnituda x dan y
%  untuk menjalankan  classification dari I.
%
%   magntuda filter Savitzky-Golay .
%
%   I matrik dari piksel  grayscale gambar.
%
%   selalu melihat SGOLAY, SGOLAYFILT, FEATURES


% gunakan k=2 and F=5
F=5;
%s = fliplr(vander(-(F-1)./2:(F-1)./2));
%S = s(:,1:k+1);   %  matrix
%M = (S'*S)\(S');
M = ...
[   -3/35        12/35        17/35        12/35
-3/35
    -1/5         -1/10          0           1/10
1/5
     1/7         -1/14        -1/7         -1/14
1/7             ];

Iav_x = mean(I);
Iav_y = mean(I');

Iav_xb = buffer(Iav_x,F,F-1,'nodelay');
Iav_yb = buffer(Iav_y,F,F-1,'nodelay');

ax = M*Iav_xb;
ay = M*Iav_yb;

a2x = ax(3,:);
a2y = ay(3,:);

Dx = sum(abs(a2x));
Dy = sum(abs(a2y));
```

```
function gc = graddir(I)
%GRADDIR kakulasi dari cardinal gradient magnituda dari
postal gambar.
%   [GC] = GRADDIR(I) penjumlahan gradient magnituda
untuk menjalankan
%   classifikasi dari I.
%
%   Gradien vektorsdadlah gabungan DIFF dan ATAN2
fungsi.
%
%   I matrik dari piksel  grayscale gambar
%
%   selalu melihat DIFF, ATAN2, FEATURES


sizeI = size(I);

I = double(I);

h=diff(I);
h=h(:,1:end-1);

v=diff(I');
v=v(:,1:end-1)';

theta = atan2(v,h);

mag = (h.^2 + v.^2).^.5;

gc = sum(sum(((theta>=(-13*pi/24) & theta<(-11*pi/24))
| (theta>=(-pi/24) & theta<(pi/24)) |
(theta>=(11*pi/24) & theta<(13*pi/24)) | (theta<(-
23*pi/24)) | (theta>=(23*pi/24))).*mag));
```

```
function new_CI = L_classifier(m, L, inputCI, image,
blocksize)
CI = inputCI;

update = 0;

% pada bagian L-list buat dan blok dari yang bukan clas
blocks setetah
% diperbaharui
% L masukan informasi tentang  blok dari yang bukan
blok dalam kolum 3, 4, 5 and 6.

while (size(L,1) ~= 0 & m > 0)

   % jalankan semua elemen  L-list
   for g = 1:size(L,1)

   m = 0;

   current_row  = L(g, 1);
   current_col  = L(g, 2);

   top = L(g,3);
   bottom = L(g,4);
   left = L(g,5);
   right = L(g,6);

   % jika diperbaharui, maka statu L-list dan
perbaharui = 1

   if (top ~= -1)
   if (CI(current_row-1, current_col,2) ~= top)
      L(g,3) = CI(current_row-1, current_col,2);
    update = 1;
   end
    end

   if (bottom ~= -1)
   if (CI(current_row+1, current_col,2) ~= bottom)
      L(g,4) = CI(current_row+1, current_col,2);
    update = 1;
   end
    end

   if (left ~= -1)
   if (CI(current_row, current_col-1,2) ~= left)
      L(g,5) = CI(current_row, current_col-1,2);
    update = 1;
```

```
   end
    end

   if (right ~= -1)
   if (CI(current_row, current_col+1,2) ~= right)
      L(g,6) = CI(current_row, current_col+1,2);
    update = 1;
   end
   end
end

h = 1;

% apakah ada perbaharuan

 while (h <= size(L,1) & update == 1)
   current_row  = L(h, 1);
   current_col  = L(h, 2);

   % mencoba blok
   classify_result = microclassifier(CI, current_row,
current_col,image, blocksize);

   if (classify_result ~= 4)

      % mengatur CI utuk calas

      [mn, std_dev] = meanSD(current_row, current_col,
blocksize, image);
      [b1, b2] =
bivalue2(image(current_row:current_row+blocksize-1,
current_col:current_col+blocksize-1));
      CI(current_row, current_col,2) = classify_result;
      CI(current_row, current_col, 3) = mn;
      CI(current_row, current_col, 4) = std_dev;
      CI(current_row, current_col, 5) = b1;
      CI(current_row, current_col, 6) = b2;


        % jika diketahui, rubah dari L-list

      if (h ~= length(L))
         L = [L(1:h-1,:); L(h+1:size(L,1),:)];
      else
         L = L(1:(size(L,1)-1),:);
      end

      m = m + 1;
```

```
   end
   h = h + 1;
end

end

new_CI = CI;
```

```
% kalkulasi dan SD untuk blok
function [m,s]=meanSD(i,j, blocksize, inputimage)

matrix=inputimage((i-1)*blocksize+1:i*blocksize, (j-
1)*blocksize+1:j*blocksize);

temp=reshape(matrix, 1, blocksize^2);

m=mean(double(temp));

s=std(double(temp));
```

```
function C = postalseg(image)
%POSTALSEG Blok segmentasi dan clasifikasi daripostal
gambar.
%   C = POSTALSEG(IMAGE)
%   katagori: gambar, text, and background.  matrix C
ditahan
%   clasifikasi dari piksel dari IMAGE.  isi C
mengikuti:
%
%   1 = gambar
%   2 = text
%   3 = background
%   4 = undetermined
%
%   IMAGE dari file nama  grayscale gambar format yang
mendukung
%   IMREAD.  IMAGE harus memiliki dimensi 64x64.
%
%   lihat IMREAD


tic
blocksize = 32;

% dapat piksel dalam matrik
% Tambahan Dahana

image='1831.jpg'

pause

I = imread(image)  % tidak termasuk yang ditambahkan
pause
image=rgb2gray(I);
bantu=image(1:192,1:192);
image=bantu;

% Batas tambahan Dahana


% kemampuan 1
CI = firstpass(I);

% performa context-based clasifikasi
while (blocksize >= 16)
    CI = splitCI(CI);

    [L, m, CIpui] = step12(CI, blocksize, I);
```

```
    CI = L_classifier(m, L, CIpui, I, blocksize);

    blocksize = blocksize/2;
end

% buat keluaran matrix harus one-to-one dari masukan
gambar
C = uint8(CI(:,:,2));
C = splitCI(C);
C = splitCI(C);
C = splitCI(C);
C = splitCI(C);

toc

% tampilkan
figure(1);
colormap(gray);
imagesc(I);
[sr,sc] = size(I);
hold on;
for i=1:sc/64-1
    plot([i*64 i*64], [2 sr-2]);
end
for i=1:sr/64-1
    plot([2 sc-2], [i*64 i*64]);
end
hold off;
axis image;
axis off;

figure(2)
imagesc(C);
hold on;
for i=1:sc/64-1
    plot([i*64 i*64], [2 sr-2]);
end
for i=1:sr/64-1
    plot([2 sc-2], [i*64 i*64]);
end
hold off;
axis image;
axis off;
```

```
function NewCI = splitCI(CI)
%SPLITCI Splits adalah  CI matrix
%
%   lihat POSTALSEG


[a,b,c]=size(CI);

NewCI = zeros(2*a,2*b,c);

NewCI(1:2:end-1,1:2:end-1,:) = CI(:,:,:);
NewCI(2:2:end,1:2:end-1,:) = CI(:,:,:);
NewCI(1:2:end-1,2:2:end,:) = CI(:,:,:);
NewCI(2:2:end,2:2:end,:) = CI(:,:,:);
```

```
function [mu,sigma,alpha] = activepixels(I,k)
%ACTIVEPIXELS
%   [MU,SIGMA,ALPHA] = ACTIVEPIXELS(I,k) standar
deviasi
%   dan isi  piksel untuk menjalankan  dari I.
%
%   ALPHA adalah isi piksels dengan intensitas < MU-
k*SIGMA.
%
%   I  matri piksel grayscale gambar.
%
%  lihat FEATURES


sizeI = size(I);

%membuat vektor dari isi piksel
vI = double(reshape(I,sizeI(1)*sizeI(2),1));

%kalkulasi
mu = mean(vI);

%kalkulasi sandar devisiasi
sigma = std(vI);

%kalkulasi dari piksel aktif
alpha = sum(vI<(mu-k*sigma));
```

```
% cek jika  CLASIFIKAS neighbor
% kembali 1 jika clasifikasi neighbor

function Result = checkAdjNeighbor(x,y,CI)
[a,b,c]=size(CI);


   Result=0;

if ((x+1)<=a)
   if (CI(x+1,y,2)~=4)
      Result=1;
   end
end

if ((y+1)<=b)
   if (CI(x,y+1,2)~=4)
      Result=1;
   end
end

if ((x-1)>=1)
   if (CI(x-1,y,2)~=4);
      Result=1;
   end
end

if ((y-1)>=1)
   if (CI(x,y-1,2)~=4)
      Result=1;
   end
end
```

```
% gandakan CI matrik untuk blok jika bukan clasifikas
neighbors

function partialList=createList(i,j,CI)

[a,b,c]=size(CI);

if ((i+1)<=a)
   w=CI(i+1,j,2);
else
   w=-1;
end

if ((i-1)>=1)
   x=CI(i-1,j,2);
else
   x=-1;
end

if ((j-1)>=1)
   y=CI(i,j-1,2);
else
   y=-1;
end


if (j+1)<=b
   z=CI(i,j+1,2);
else
   z=-1;
end

partialList=[x w y z];
```

```
function A = features(I)
%FEATURES kalkulasi postal gambar blok clasifikasi
benar.
%   A = FEATURES(I) kalkulasi 9 untuk menjalankan
clasifikasi dari I.
%   9 ketentuan mengikuti:
%
%   A.mu     = isi piksel intensitas
%   A.sigma  = standar devisiasi dari piksel
intensitaes
%   A.alpha  = isi pixels
%   A.Dx     = penjumlahan dari magnituda dalam x
%   A.Dy     = penjumlahan dari magnituda dalam y
%   A.gc     = penjumlahan dari vector magnituda
%
%   Iadalah matrik dari piksel  grayscale gambar
%   lihai CLASSIFY64, ACTIVEPIXELS, DXDY, GRADDIR

%merupakan, standar deviasi, dan  piksels
[A.mu A.sigma A.alpha] = activepixels(I,1.8);

%hasil Dx dan Dy untuk polynom order 2 dan bagian
tangan 5
[A.Dx A.Dy] = DxDy(I);

%enjulahkan gradient vector dalam 4 cardinal
A.gc = graddir(I);
```

```
function CI = firstpass(I)
%FIRSTPASS segmen masukan dan clasifikasi dari gambar.
%   CI = FIRSTPASS(I) performa pertama dari klasifikasi
dari
%   katagori: gambar, text, background, or
undetermined.  This classification
%   is stored in CI(:,:,2) as follows:
%
%   1 = gambar
%   2 = text
%   3 = background
%   4 = undetermined
%
%   CI(:,:,3) masukan blok, CI(:,:,4) masukan sandar
devisiasi , dan CI(:,:,[5,6]) masukan bi-intensity .

%   lihat POSTALSEG, CLASSIFY64, STEP12, L_CLASSIFIER,
IMREAD

% kelaskan semua gambar dan semua matrix CI

% Tambahan Dahana
I=I(1:192,1:192,:);

[SR,SC] = size(I)
A = 1;
for i=1:64:SR
    B = 1;
    for j=1:64:SC
       % dapat gambar
       block = I([i:i+63],[j:j+63]);

       % pertamadapat masukan clasifikasi, artinya, dan
standar deviasi
       [CI(A,B,2) CI(A,B,3) CI(A,B,4)] =
classify64(block);

       % ambil isi bi-intensity
       [CI(A,B,5),CI(A,B,6)] = bivalue2(block);

       B = B + 1;
   end
   A = A + 1;
end

%figure(2);
%gambarc(CI(:,:,2));
%axis gambar;
```

```
function xss = hangsegment(parseimage, filename,
blocksize2)

% segmen gambar

blocksize = blocksize2;

imageA=imread(parseimage,'png');
imageC=double(imageA);
%imageB=clean(imageC);
imageB = imageC;
a = size(imageB,1);
b = size(imageB,2);

row = a./blocksize2;
col = b./blocksize2;

initCI = zeros(row,col);

figure(1)
imagesc(imageB)
colormap(gray)
hold on

%lukisan grid
   for kq = 1:blocksize2:b
      plot([kq kq], [1 a],'b')
   end

   for nq = 1:blocksize2:a
      plot([1 b], [nq nq],'b')

    end

   figure(2)
   imagesc(initCI)

kinp = 0;
type = 3;

figure(1)
while (kinp ~= 4)

kinp = menu('Options', 'gambar', 'Text', 'Quit');
figure(1)

switch kinp
case 1,
```

```
   type = 1
case 2,
   type = 2
case 3,
   save filename initCI
   xss = [];
   return
end

% get coordinates
figure(1)
[x,y] = ginput(2);

x = round(x);
y = round(y);

colcel = ceil(x./blocksize);
rowcel = ceil(y./blocksize);


% set tipe
for u = colcel(1):colcel(2)
   for uq = rowcel(1):rowcel(2)
      initCI(uq,u) = type;
   end
end


figure(2)
imagesc(initCI)
pause

figure(1)
end


xss = [];
```

```
function x = microclassifier(CI, row, col, image,
blocksize)

matrix=image((row-1)*blocksize+1:row*blocksize, (col-
1)*blocksize+1:col*blocksize);


% tes Background
tolerance_bk = 0.1 * 255;

if (CI(row,col,1) == inf) | (abs((mean(mean(matrix)) -
255)) <= tolerance_bk)
   x = 3;
   return
end

% set row,col dari  block tetangga
adj_col = [col col col-1 col+1];
adj_row = [ row-1 row+1 row row];

% text positive - 1
% gambar positive - 2
type_relative = [];

tolerance_text = 80;
tolerance_image = 50;

% jalankan semua 4 tetangga
for n = 1:4

if ((adj_row(n) > 0) & (adj_col(n) > 0) &(adj_col(n) <
size(CI,2)) & (adj_row(n) < size(CI,1)))

   % jika tetangga teks
   if (CI(adj_row(n),adj_col(n),2) == 2)
      if ((CI(row,col,5) ~= -1) & (CI(row,col,6) ~= -1)
& ...
         (abs((CI(row,col,5) - CI(adj_row(n),
adj_col(n), 5))) <= tolerance_text) & ...
         (abs((CI(row,col,6) - CI(adj_row(n),
adj_col(n), 6)))  <= tolerance_text))

       type_relative = [type_relative 2];
      end

   % jika tetangga gambar
    elseif (CI(adj_row(n),adj_col(n),2) == 1)
```

```
    if (abs((CI(row,col,3)- CI(adj_row(n),
adj_col(n),3))) <= tolerance_image)
       type_relative = [type_relative 1];
    end

   % tidak masuk klas
    else
    type_relative = [type_relative 4];
    end

else
   type_relative = [type_relative 0];
end


x = 4;

for u = 1:length(type_relative)
   if (type_relative(u) == 1)
      x = 1;
   end
end

for u = 1:length(type_relative)
   if (type_relative(u) == 2)
      x = 2;
   end
end



end
```

```
function [List,m, CInew] = step12(CItemp,blocksize,
originalImage)
m=0;
List=[];

CI = CItemp;
[a, b,c]=size(CI);

for i=1:a
   for j=1:b
      if CI(i,j,2)==4 % 4 representasi tidak berkelas
      AdjNeighbor=checkAdjNeighbor(i,j,CI); %cek jika
tidak clasifikasiblock jika clasifikasi neighbors

            if AdjNeighbor==0 %jika tidak clasifikasi
neighbors, atambahkan L
            List=[List; [i j createList(i,j,CI)]];
        else

         classtype=microclassifier(CI,
i,j,originalImage, blocksize);

            if classtype==4 %if microclassifier cannot
classify block
                List=[List; [i j createList(i,j,CI)]];
            else
               %clasifikasi blok,  CI matrik
               [bivalue_1, bivalue_2] =
bivalue2(originalImage((i-1)*blocksize +1:i*blocksize,
(j-1)*blocksize+1:j*blocksize));

               [tt, ttt] =
meanSD(i,j,blocksize,originalImage);
               temptemp = cat(3, -1, classtype, tt,ttt,
bivalue_1, bivalue_2);

               CI(i,j,:)= temptemp;

               m=m+1;

          end
         end
      end
   end
end
CInew = CI;
```

```
clear;
close all;
clc;

tic
blocksize = 32;

% get pixel value matrix

% Tambahan Dahana

image='1831.jpg'

pause

I = imread(image)  % tidak termasuk yang ditambahkan
pause
% % image=rgb2gray(I);
% image=I;
% bantu=image(1:192,1:192);
% image=bantu;

% Batas tambahan Dahana


% perfoma pertama
CI = firstpass(I);

% perfoma context-based
while (blocksize >= 16)
    CI = splitCI(CI);

    [L, m, CIpui] = step12(CI, blocksize, I);

    CI = L_classifier(m, L, CIpui, I, blocksize);

    blocksize = blocksize/2;
end

% buat keluaran matrik harus one-to-one koresponden
gambar
C = uint8(CI(:,:,2));
C = splitCI(C);
C = splitCI(C);
C = splitCI(C);
C = splitCI(C);

toc
```

```
% tampilkan
figure(1);
colormap(gray);
imagesc(I);
[sr,sc] = size(I);
hold on;
for i=1:sc/64-1
    plot([i*64 i*64], [2 sr-2]);
end
for i=1:sr/64-1
    plot([2 sc-2], [i*64 i*64]);
end
hold off;
axis image;
axis off;

figure(2)
imagesc(C);
hold on;
for i=1:sc/64-1
    plot([i*64 i*64], [2 sr-2]);
end
for i=1:sr/64-1
    plot([2 sc-2], [i*64 i*64]);
end
hold off;
axis image;
axis off;
```