

LAMPIRAN A
LISTING PROGRAM

```

clear;
close all;
clc;

% -----%
% RSM
% -----%
% -----%
% Gambar masukan
% -----%
x_all=imread('tujuh','jpg');

% -----%
% Mencari ukuran gambar
% -----%
[brs kol ruang]=size(x_all);

% -----%
% Memecah gambar menjadi beberapa komponen spektral (warna)
% -----%
x_red=x_all(:,:,1);
x_green=x_all(:,:,2);
x_blue=x_all(:,:,3);

% -----%
% Mengubah tipe data dari uint8 menjadi double
% -----%
x_red=double(x_red);
x_green=double(x_green);
x_blue=double(x_blue);

% -----%
% Mengubah masing-masing komponen spektral ke domain frekuensi
% -----%
fft_x_red=fft2(x_red);
fft_x_green=fft2(x_green);
fft_x_blue=fft2(x_blue);

% -----%
% Menguji komponen real
% -----%
for j=1:brs
    for k=1:kol
        fft_x_real_red(j,k)=real(fft_x_red(j,k));
        fft_x_real_green(j,k)=real(fft_x_green(j,k));
        fft_x_real_blue(j,k)=real(fft_x_blue(j,k));
    end;
end;

```

```

end;

% -----%
% Hapus variabel bantu
% -----%
clear j k

% -----%
% Hitung autokorelasi
% -----%
autokor_x_red=xcorr2(fft_x_real_red);
autokor_x_green=xcorr2(fft_x_real_green);
autokor_x_blue=xcorr2(fft_x_real_blue);

% -----%
% Mencari nilai eigen dan vektor eigen
% -----%
[vek_x_red eig_x_red]=eig(autokor_x_red);
[vek_x_green eig_x_green]=eig(autokor_x_green);
[vek_x_blue eig_x_blue]=eig(autokor_x_blue);

% -----%
% Mencari nilai eigen dari matriks eig_x ukuran brs x kol dan mengubah
% menjadi sebuah array
% -----%
m=1;
for j=1:size(eig_x_red,1)
    for k=1:size(eig_x_red,2)
        if j==k & m<=size(eig_x_red,1)
            array_eig_x_red(m)=eig_x_red(j,k);
            array_eig_x_green(m)=eig_x_green(j,k);
            array_eig_x_blue(m)=eig_x_blue(j,k);
        end;
    end;
    m=m+1;
end;

% -----%
% Hapus variabel bantu
% -----%
clear j k m

% -----%
% Mengambil nilai eigen yang real saja dan sekaligus mengambil vektor eigen
% yang bersesuaian dengan nilai eigen tersebut
% -----%

```

```

% -----%
% (1) Spektral pertama (RED)
% -----%
m=1;
for j=1:length(array_eig_x_red)
    if imag(array_eig_x_red(j))==0
        real_array_eig_x_red(m)=real(array_eig_x_red(j));
        vek_real_array_eig_x_red(1:length(array_eig_x_red),m)=...
            vek_x(1:length(array_eig_x_red),j);
        m=m+1;
    end;
end;

% -----%
% Hapus variabel bantu
% -----%
clear j m

% -----%
% (2) Spektral kedua (GREEN)
% -----%
m=1;
for j=1:length(array_eig_x_green)
    if imag(array_eig_x_green(j))==0
        real_array_eig_x_green(m)=real(array_eig_x_green(j));
        vek_real_array_eig_x_green(1:length(array_eig_x_green),m)=...
            vek_x(1:length(array_eig_x_green),j);
        m=m+1;
    end;
end;

% -----%
% Hapus variabel bantu
% -----%
clear j m

% -----%
% (3) Spektral ketiga (BLUE)
% -----%
m=1;
for j=1:length(array_eig_x_blue)
    if imag(array_eig_x_blue(j))==0
        real_array_eig_x_blue(m)=real(array_eig_x_blue(j));
        vek_real_array_eig_x_blue(1:length(array_eig_x_blue),m)=...
            vek_x(1:length(array_eig_x_blue),j);
        m=m+1;
    end;
end;

```

```

end;

% -----%
% Hapus variabel bantu
% -----%
clear j m

% -----%
% Menggabungkan nilai eigen dengan vektor eigen yang bersesuaian
% -----%
gabung_array_x_red=[real_array_eig_x_red;vek_real_array_eig_x_red];
gabung_array_x_green=[real_array_eig_x_green;vek_real_array_eig_x_green];
gabung_array_x_blue=[real_array_eig_x_blue;vek_real_array_eig_x_blue];

p1=gabung_array_x_red;
p2=gabung_array_x_green;
p3=gabung_array_x_blue;

x1=p1(1,:);
x2=p2(1,:);
x3=p3(1,:);

[brs1 kol1]=size(p1);
urut_atas_1=sort(p1(1,:));
y1(1,:)=urut_atas_1;

[brs2 kol2]=size(p2);
urut_atas_2=sort(p2(1,:));
y2(1,:)=urut_atas_2;

[brs3 kol3]=size(p3);
urut_atas_3=sort(p3(1,:));
y3(1,:)=urut_atas_3;

% -----%
% Mencari urutan yang sesuai (RED)
% -----%
for j=1:length(y1)
    [baris_1 indeks_1(j)]=find(y1(1,j)==x1);
end;

% -----%
% Hapus variabel bantu
% -----%
clear j

% -----%

```

```

% Mencari urutan yang sesuai (GREEN)
% -----%
for j=1:length(y2)
    [baris_2 indeks_2(j)]=find(y2(1,j)==x2);
end;

% -----%
% Hapus variabel bantu
% -----%
clear j

% -----%
% Mencari urutan yang sesuai (BLUE)
% -----%
for j=1:length(y3)
    [baris_3 indeks_3(j)]=find(y3(1,j)==x3);
end;

% -----%
% Hapus variabel bantu
% -----%
clear j

% -----%
% Mencari urutan secara ascending order (RED)
% -----%
for j=1:kol1
    z1(:,j)=p1(:,indeks_1(j));
end;

% -----%
% Hapus variabel bantu
% -----%
clear j

% -----%
% Mencari urutan secara ascending order (GREEN)
% -----%
for j=1:kol2
    z2(:,j)=p2(:,indeks_2(j));
end;

% -----%
% Hapus variabel bantu
% -----%
clear j

```

```

% -----%
% Mencari urutan secara ascending order (BLUE)
% -----%
for j=1:kol3
    z3(:,j)=p3(:,indeks_3(j));
end;

% -----%
% Hapus variabel bantu
% -----%
clear j

% -----%
% Mengubah urutan menjadi descending order
% -----%
zz1=fliplr(z1);
zz2=fliplr(z2);
zz3=fliplr(z3);

[brs_zz1 kol_zz1]=size(zz1);
[brs_zz2 kol_zz2]=size(zz2);
[brs_zz3 kol_zz3]=size(zz3);

% -----%
% Mengambil vektor eigen saja berdasarkan nilai eigennya (RED)
% -----%
m=1;
for j=1:kol_zz1
    if zz1(1,j)>0
        vek_final_1(1:brs_zz1-1,m)=zz1(2:brs_zz1,j);
        end;
        m=m+1;
    end;

% -----%
% Hapus variabel bantu
% -----%
clear j m

% -----%
% Mengambil vektor eigen saja berdasarkan nilai eigennya (GREEN)
% -----%
m=1;
for j=1:kol_zz2
    if zz2(1,j)>0
        vek_final_2(1:brs_zz2-1,m)=zz2(2:brs_zz2,j);
        end;
end;

```

```

    m=m+1;
end;

% -----%
% Hapus variabel bantu
% -----%
clear j m

% -----%
% Mengambil vektor eigen saja berdasarkan nilai eigennya (BLUE)
% -----%
m=1;
for j=1:kol_zz3
    if zz3(1,j)>0
        vek_final_3(1:brs_zz3-1,m)=zz3(2:brs_zz3,j);
    end;
    m=m+1;
end;

% -----%
% Hapus variabel bantu
% -----%
clear j m

% -----%
% Mengembalikan ke domain spasial
% -----%
ifft_x_red=ifft2(vek_final_1);
ifft_x_green=ifft2(vek_final_2);
ifft_x_blue=ifft2(vek_final_3);

gambar_rekon(:,:,1)=ifft_x_red;
gambar_rekon(:,:,2)=ifft_x_green;
gambar_rekon(:,:,3)=ifft_x_blue;
[brs_fin kol_fin]=size(gambar_rekon);

% -----%
% Perhitungan PSNR
% -----%
x_all=reshape(x_all,1,brs_fin*kol_fin);
gambar_rekon=reshape(gambar_rekon,1,brs_fin*kol_fin);
MSE_rekon=sum((x_all-gambar_rekon).^2);
PSNR_rekon=10.*log10((255.^2)/(MSE_rekon/(brs_fin*kol_fin)));
disp(['Nilai SNR akhir setelah rekonstruksi = ',num2str(PSNR_rekon),' dB']);
disp(' ');

figure;imshow(uint8(gambar_rekon));

```



```

% -----%
% CSM
% -----%

clear;
close all;
clc;

% -----%
% % Gambar masukan
% -----%
x_all=imread('tujuh','jpg');

% -----%
% Mencari ukuran gambar
% -----%
[brs kol ruang]=size(x_all);

% -----%
% Memecah gambar menjadi beberapa komponen spektral (warna)
% -----%
x_red=x_all(:,:,1);
x_green=x_all(:,:,2);
x_blue=x_all(:,:,3);

% -----%
% Mengubah tipe data dari uint8 menjadi double
% -----%
x_red=double(x_red);
x_green=double(x_green);
x_blue=double(x_blue);

% -----%
% Mengubah masing-masing komponen spektral ke domain frekuensi
% -----%
fft_x_red=fft2(x_red);
fft_x_green=fft2(x_green);
fft_x_blue=fft2(x_blue);

% -----%
% Menguji komponen imag
% -----%
for j=1:brs

```

```

for k=1:kol
    fft_x_imag_red(j,k)=imag(fft_x_red(j,k));
    fft_x_imag_green(j,k)=imag(fft_x_green(j,k));
    fft_x_imag_blue(j,k)=imag(fft_x_blue(j,k));
end;
end;

% -----%
% Hapus variabel bantu
% -----%
clear j k

% -----%
% Hitung autokorelasi
% -----%
autokor_x_red=xcorr2(fft_x_imag_red);
autokor_x_green=xcorr2(fft_x_imag_green);
autokor_x_blue=xcorr2(fft_x_imag_blue);

% -----%
% Mencari nilai eigen dan vektor eigen
% -----%
[vek_x_red eig_x_red]=eig(autokor_x_red);
[vek_x_green eig_x_green]=eig(autokor_x_green);
[vek_x_blue eig_x_blue]=eig(autokor_x_blue);

% -----%
% Mencari nilai eigen dari matriks eig_x ukuran brs x kol dan mengubah
% menjadi sebuah array
% -----%
m=1;
for j=1:size(eig_x_red,1)
    for k=1:size(eig_x_red,2)
        if j==k & m<=size(eig_x_red,1)
            array_eig_x_red(m)=eig_x_red(j,k);
            array_eig_x_green(m)=eig_x_green(j,k);
            array_eig_x_blue(m)=eig_x_blue(j,k);
        end;
    end;
    m=m+1;
end;

% -----%
% Hapus variabel bantu
% -----%
clear j k m

```

```

% -----%
% Mengambil nilai eigen yang real saja dan sekaligus mengambil vektor eigen
% yang bersesuaian dengan nilai eigen tersebut
% -----%

% -----%
% (1) Spektral pertama (RED)
% -----%
m=1;
for j=1:length(array_eig_x_red)
    if imag(array_eig_x_red(j))==0
        real_array_eig_x_red(m)=real(array_eig_x_red(j));
        vek_real_array_eig_x_red(1:length(array_eig_x_red),m)=...
            vek_x(1:length(array_eig_x_red),j);
        m=m+1;
    end;
end;

% -----%
% Hapus variabel bantu
% -----%
clear j m

% -----%
% (2) Spektral kedua (GREEN)
% -----%
m=1;
for j=1:length(array_eig_x_green)
    if imag(array_eig_x_green(j))==0
        real_array_eig_x_green(m)=real(array_eig_x_green(j));
        vek_real_array_eig_x_green(1:length(array_eig_x_green),m)=...
            vek_x(1:length(array_eig_x_green),j);
        m=m+1;
    end;
end;

% -----%
% Hapus variabel bantu
% -----%
clear j m

% -----%
% (3) Spektral ketiga (BLUE)
% -----%
m=1;
for j=1:length(array_eig_x_blue)

```

```

if imag(array_eig_x_blue(j))==0
    real_array_eig_x_blue(m)=real(array_eig_x_blue(j));
    vek_real_array_eig_x_blue(1:length(array_eig_x_blue),m)=...
        vek_x(1:length(array_eig_x_blue),j);
    m=m+1;
end;
end;

% -----%
% Hapus variabel bantu
% -----%
clear j m

% -----%
% Menggabungkan nilai eigen dengan vektor eigen yang bersesuaian
% -----%
gabung_array_x_red=[real_array_eig_x_red;vek_real_array_eig_x_red];
gabung_array_x_green=[real_array_eig_x_green;vek_real_array_eig_x_green];
gabung_array_x_blue=[real_array_eig_x_blue;vek_real_array_eig_x_blue];

p1=gabung_array_x_red;
p2=gabung_array_x_green;
p3=gabung_array_x_blue;

x1=p1(1,:);
x2=p2(1,:);
x3=p3(1,:);

[brs1 kol1]=size(p1);
urut_atas_1=sort(p1(1,:));
y1(1,:)=urut_atas_1;

[brs2 kol2]=size(p2);
urut_atas_2=sort(p2(1,:));
y2(1,:)=urut_atas_2;

[brs3 kol3]=size(p3);
urut_atas_3=sort(p3(1,:));
y3(1,:)=urut_atas_3;

% -----%
% Mencari urutan yang sesuai (RED)
% -----%
for j=1:length(y1)
    [baris_1 indeks_1(j)]=find(y1(1,j)==x1);
end;

```

```

% -----%
% Hapus variabel bantu
% -----%
clear j

% -----%
% Mencari urutan yang sesuai (GREEN)
% -----%
for j=1:length(y2)
    [baris_2 indeks_2(j)]=find(y2(1,j)==x2);
end;

% -----%
% Hapus variabel bantu
% -----%
clear j

% -----%
% Mencari urutan yang sesuai (BLUE)
% -----%
for j=1:length(y3)
    [baris_3 indeks_3(j)]=find(y3(1,j)==x3);
end;

% -----%
% Hapus variabel bantu
% -----%
clear j

% -----%
% Mencari urutan secara ascending order (RED)
% -----%
for j=1:kol1
    z1(:,j)=p1(:,indeks_1(j));
end;

% -----%
% Hapus variabel bantu
% -----%
clear j

% -----%
% Mencari urutan secara ascending order (GREEN)
% -----%
for j=1:kol2
    z2(:,j)=p2(:,indeks_2(j));
end;

```

```

% -----%
% Hapus variabel bantu
% -----%
clear j

% -----%
% Mencari urutan secara ascending order (BLUE)
% -----%
for j=1:kol3
    z3(:,j)=p3(:,indeks_3(j));
end;

% -----%
% Hapus variabel bantu
% -----%
clear j

% -----%
% Mengubah urutan menjadi descending order
% -----%
zz1=fliplr(z1);
zz2=fliplr(z2);
zz3=fliplr(z3);

[brs_zz1 kol_zz1]=size(zz1);
[brs_zz2 kol_zz2]=size(zz2);
[brs_zz3 kol_zz3]=size(zz3);

% -----%
% Mengambil vektor eigen saja berdasarkan nilai eigennya (RED)
% -----%
m=1;
for j=1:kol_zz1
    if zz1(1,j)>0
        vek_final_1(1:brs_zz1-1,m)=zz1(2:brs_zz1,j);
        end;
        m=m+1;
    end;

% -----%
% Hapus variabel bantu
% -----%
clear j m

% -----%
% Mengambil vektor eigen saja berdasarkan nilai eigennya (GREEN)

```

```

% -----%
m=1;
for j=1:kol_zz2
    if zz2(1,j)>0
        vek_final_2(1:brs_zz2-1,m)=zz2(2:brs_zz2,j);
    end;
    m=m+1;
end;

% -----%
% Hapus variabel bantu
% -----%
clear j m

% -----%
% Mengambil vektor eigen saja berdasarkan nilai eigennya (BLUE)
% -----%
m=1;
for j=1:kol_zz3
    if zz3(1,j)>0
        vek_final_3(1:brs_zz3-1,m)=zz3(2:brs_zz3,j);
    end;
    m=m+1;
end;

% -----%
% Hapus variabel bantu
% -----%
clear j m

% -----%
% Mengembalikan ke domain spasial
% -----%
ifft_x_red=ifft2(vek_final_1);
ifft_x_green=ifft2(vek_final_2);
ifft_x_blue=ifft2(vek_final_3);

gambar_rekon(:,:,1)=ifft_x_red;
gambar_rekon(:,:,2)=ifft_x_green;
gambar_rekon(:,:,3)=ifft_x_blue;
[brs_fin kol_fin]=size(gambar_rekon);

% -----%
% Perhitungan PSNR
% -----%
x_all=reshape(x_all,1,brs_fin*kol_fin);
gambar_rekon=reshape(gambar_rekon,1,brs_fin*kol_fin);

```

```

MSE_rekon=sum((x_all-gambar_rekon).^2);
PSNR_rekon=10.*log10((255.^2)./(MSE_rekon/(brs_fin*kol_fin)));
disp(['Nilai SNR akhir sesudah rekonstruksi = ',num2str(PSNR_rekon),' dB']);
disp(' ');

figure;imshow(uint8(gambar_rekon));

% -----%
% HUFFMAN
% -----%

if nargin<1
    error('huffcode: see help.')
end
if nargin<2
    Display = 0;
end
if (Display ~= 1)
    Display = 0;
end

N=length(HL);
L=max(HL);
HK=zeros(N,L);
[HLs,HLi] = sort(HL);
Code=zeros(1,L);
for n=1:N
    if (HLs(n)>0)
        HK(HLi(n,:),:) = Code;
        k = HLs(n);
        while (k>0)
            Code(k) = Code(k) + 1;
            if (Code(k)==2)
                Code(k) = 0;
                k=k-1;
            else
                break
            end
        end
    end
end
end

if Display
    for n=1:N
        Linje = [' Symbol ',int2str(n)];
        for i=length(Linje):15
            Linje = [Linje, ''];
        end
    end
end

```



```
end
Linje = [Linje,' gets code: '];
for i=1:HL(n)
    if (HK(n,i)==0)
        Linje = [Linje,'0'];
    else
        Linje = [Linje,'1'];
    end
end
disp(Linje);
end
end

return;
```