

Apéndice

Estos fueron los algoritmos utilizados.

```
%Division de la imagen en bloques de 8x8
function matdim8=divide8(matcom)
tdblo=8;
[rf,cf]=size(matcom);
%s
sobhor=rem(rf,8);
blohor=(rf-sobhor)/8;
sobver=rem(cf,8);
blover=(cf-sobver)/8;
for bh=1:blohor
renori=tdblo*(bh-1);
for bv=1:blover
colori=tdblo*(bv-1);
for rrr=1:8
for ccc=1:8
maenblo(rrr,ccc,bh,bv)=matcom(rrr+renori,ccc+colori);
end
end
end
end
matdim8=maenblo;
```

Transformada Discreta Coseno

```
function b=dct(a,n)
if nargin == 0,
    error('Not enough input arguments.');
```

end

```
if isempty(a)
    b = [];
    return
end
```

```
do_trans = (size(a,1) == 1);
if do_trans, a = a(:); end
```

```
if nargin==1,
    n = size(a,1);
end
m = size(a,2);
```

```
if size(a,1)<n,
    aa = zeros(n,m);
    aa(1:size(a,1),:) = a;
else
    aa = a(1:n,:);
end
```

```
ww = (exp(-i*(0:n-1)*pi/(2*n))/sqrt(2*n)).!;
ww(1) = ww(1) / sqrt(2);
```

```
if rem(n,2)==1 | ~isreal(a), % odd case
    y = zeros(2*n,m);
    y(1:n,:) = aa;
    y(n+1:2*n,:) = flipud(aa);

    yy = fft(y);
    yy = yy(1:n,:);
```

```
else % even case
    y = [ aa(1:2:n,:); aa(n:-2:2,:) ];
    yy = fft(y);
    ww = 2*ww;
end
```

```
b = ww(:,ones(1,m)) .* yy;
if isreal(a), b = real(b); end
if do_trans, b = b.!' ; end
```

Codificacion

```
function codifi_04(matbidi,r,c)
[rm,cm]=size(matbidi);
%%
fid = fopen('eje_04.bin','wb');
fwrite(fid,cm,'ubit8');
fwrite(fid,r,'ubit8');
fwrite(fid,c,'ubit8');
if cm==0
%%
else
%%
dctemp=matbidi(:,1);
liminfe=min(dctemp);
limsupe=max(dctemp);
rango=deterran02(liminfe,limsupe);%
iden=str2num(rango);
fwrite(fid,iden,'ubit4');%%
bitsreq=strcat('bit',rango);
for m1=1:rm
    fwrite(fid,matbidi(m1,1),bitsreq);
end
%%
if cm>1
    matbidi(:,1)=0;
    if cm<=36
        aux_03(matbidi,2,cm,fid);
    else
        mataux=matbidi;
        for tp=37:cm
            mataux(:,tp)=0;
        end
        aux_03(mataux,2,36,fid);
        %.
        %%.
        for t2=2:36 %%
            matbidi(:,t2)=0;
        end
        aux_03(matbidi,37,cm,fid);
        %%.
    end
end
end
end
fclose(fid);
```

Archivo auxiliar para la codificacion

```
function aux_03(matbidix,rini,rfin,fid01)
bidiInve=matbidix.';
[ri,ci]=size(bidiInve);
minimo=min(bidiInve);
maximo=max(bidiInve);
%%AC
for t=1:ci
    liminfe=minimo(t);
    limsupe=maximo(t);
    rango=deterran02(liminfe,limsupe);%
    iden=str2num(rango);
    fwrite(fid01,iden,'ubit4');%%
    bitsreq=strcat('bit',rango);
    %%
    for m2=rini:rfin
        fwrite(fid01,bidiInve(m2,t),bitsreq);
    end
    %%
end

if (li>-2)&(ls<1)
    randsal='1';
elseif (li>-3)&(ls<2)
    randsal='2';
elseif (li>-5)&(ls<4)
    randsal='3';
elseif (li>-9)&(ls<8)
    randsal='4';
elseif (li>-17)&(ls<16)
    randsal='5';
elseif (li>-33)&(ls<32)
    randsal='6';
elseif (li>-65)&(ls<64)
    randsal='7';
elseif (li>-129)&(ls<128)
    randsal='8';
elseif (li>-257)&(ls<256)
    randsal='9';
elseif (li>-513)&(ls<512)
    randsal='10';
elseif (li>-1025)&(ls<1024)
    randsal='11';
elseif (li>-2049)&(ls<2048)
    randsal='12';
else
    randsal='13';
```

Transformada Inversa coseno

```

function a = idct(b,n)
if nargin == 0,
    error('Not enough input arguments.');
```

end

```

if isempty(b),
    a = [];
    return
end
do_trans = (size(b,1) == 1);
if do_trans, b = b(:); end
    if nargin==1,
        n = size(b,1);
    end
m = size(b,2);
if size(b,1)<n,
    bb = zeros(n,m);
    bb(1:size(b,1),:) = b;
else
    bb = b(1:n,:);
end
ww = sqrt(2*n) * exp(j*(0:n-1)*pi/(2*n)).';
if rem(n,2)==1 | ~isreal(b), % odd case
    ww(1) = ww(1) * sqrt(2);
    W = ww(:,ones(1,m));
    yy = zeros(2*n,m);
    yy(1:n,:) = W.*bb;
    yy(n+2:2*n,:) = -j*W(2:n,:). *flipud(bb(2:n,:));
    y = ifft(yy);
    a = y(1:n,:);

else % even case
    ww(1) = ww(1)/sqrt(2);
    W = ww(:,ones(1,m));
    yy = W.*bb;

y = ifft(yy);

a = zeros(n,m);
a(1:2:n,:) = y(1:n/2,:);
a(2:2:n,:) = y(n-1:n/2+1,:);
end

if isreal(b), a = real(a); end
if do_trans, a = a.'; end
```

Decodificacion

```
function [maddosd,rdb,cdb]=decodifi
fid = fopen('eje_04.bin','rb');
cm=fread(fid,1,'ubit8');
R=fread(fid,1,'ubit8');
C=fread(fid,1,'ubit8');
rm=R*C;
if cm==0
    maddosd=[];
else
    %
    maddosd=zeros(rm,cm);
    %%
    %%
    clave=fread(fid,1,'ubit4');
    cad01=num2str(clave);
    bitusa=strcat('bit',cad01);
    for m1=1:rm
        maddosd(m1,1)=fread(fid,1,bitusa);
    end
    %%
    if cm>1
        if cm<=36
            temp1=maddosd.';
            [ri,ci]=size(temp1);
            %%
            for t=1:ci
                clave=fread(fid,1,'ubit4');
                cad01=num2str(clave);
                bitusa=strcat('bit',cad01);
                for m2=2:ri
                    temp1(m2,t)=fread(fid,1,bitusa);
                end
            end
            %%
            maddosd=temp1.';
        else
            temp1=maddosd.';
            [ri,ci]=size(temp1);
            %%
            for t=1:ci
                clave=fread(fid,1,'ubit4');
                cad01=num2str(clave);
                bitusa=strcat('bit',cad01);
                for m2=2:36
                    temp1(m2,t)=fread(fid,1,bitusa);
                end
            end
        end
    end
end
```

```
    end
  end
  %%
  %%
  for t=1:ci
    clave=fread(fid,1,'ubit4');
    cad01=num2str(clave);
    bitusa=strcat('bit',cad01);
    for m2=37:ri
      temp1(m2,t)=fread(fid,1,bitusa);
    end
  end
  %%
  maddosd=temp1.';
end

end
end
rdb=R;
cdb=C;
fclose(fid);
```

Realiza la union de la imagen dividida

```
function matinv8=individe8(mataconv,Re,Co)
tmblo=8;
for blore=1:Re
    reori=tmblo*(blore-1);
    for bloco=1:Co
        cori=tmblo*(bloco-1);
        for reng=1:8
            for colu=1:8
                mat_temp(reng+reori,colu+cori)=mataconv(reng,colu,blore,bloco);
            end
        end
    end
end
matinv8=mat_temp;
```

Detector de errores

```
function calderro(mdent,mdsal)
%%determinación de los errores.
%%recibe la matriz de entrada y salida
%%con ello determina errores y presenta datos
```

```
mat_error=(mdent-mdsal).^2;
ima_error=sqrt(mat_error);
[m,n]=size(mdent);
lonDentra=m*n;%%
temp=num2str(lonDentra);
cade1=["temp" ' ' 'bytes'];
%%
archi_01='eje_04.bin';
r1=dir(archi_01);
lonComp=r1.bytes;%%
temp=num2str(lonComp);
cade2=["temp" ' ' 'bytes'];

fdcom=lonDentra/lonComp;%
temp=num2str(fdcom,'%0.3f');
cade3=["temp" ' ' ': ' '1'];

cuenta=0;
for r=1:m
    for c=1:n
        cuenta=cuenta+mat_error(r,c);
    end
end
mse=cuenta/lonDentra;%
mserms=sqrt(mse);
temp=num2str(mserms,'%0.3f');
cade4=["temp" ' ' 'rms'];
psnr=10*log10((255)^2/mse);%
temp=num2str(psnr,'%0.2f');
cade5=["temp" ' ' 'dB'];
datVa=struct('posi',{cade1",cade2",...
    "cade3",cade4"...
    "cade5"});

%%
maneLo=get(gcf,'Userdata');
con01=7; %
%%
```

```

axes(manLo(con01),image(uint8(mat_error)), title('Error cuadrático');
    set(manLo(con01),'Xtick',[],'Ytick',[]);
%%
[rr,cc]=size(datVa);
for t=2:2:cc
    con01=con01+1;
    set(manLo(con01),'string',' ');
    set(manLo(con01),'string',datVa(1,t).posi);
end

```

Programa de Control

```

function dct03(accion);
%programa de control

clc;
if nargin<1
    accion='inicio';
end
if strcmp(accion,'inicio')
    posdefa=[0.1 0.1 0.75 0.75];
    colores=[0.1 0.2 0.3]; %aquí 0.8 0.9 0.8
    man_fig=figure( ...
        'Name',' Compresion empleando DCT', ...
        'NumberTitle','off','color',colores,...
        'pointer','arrow','Units','normalized','position',posdefa);

    punSuper=0.93;
    %
    izq=0.785;
    anchBot=0.175;
    altBot=0.06;
    %
    espacio=0.025;

    frmBorde=0.02; frmInferior=0.05; frmAltura= 0.5;
    posY=punSuper-frmAltura;
    Posfrm=[izq-frmBorde posY anchBot+2*frmBorde frmAltura+2*frmBorde];
    h=icontrol( ...
        'Style','frame', ...
        'Units','normalized', ...
        'Position',Posfrm, ...
        'BackgroundColor',[0.1 0.2 0.3]); %0.85 0.89 0.9
    labelStr='Lena';
    callbackStr='dct03("lena256")';

```

```
posY=punSuper-altBot-espacio;
gene=uicontrol( ...
'Style','pushbutton', ...
'Units','normalized', ...
'Position',[izq posY anchBot altBot], ...
'String',labelStr,'callback',callbackStr);
%
labelStr='Matrix';
callbackStr='dct03("Matrix")';
posY=punSuper-2*(altBot+espacio);
gene=uicontrol( ...
'Style','pushbutton', ...
'Units','normalized', ...
'Position',[izq posY anchBot altBot], ...
'String',labelStr,'callback',callbackStr);
%
labelStr='Camara';
callbackStr='dct03("camera")';
posY=punSuper-3*(altBot+espacio);
gene=uicontrol( ...
'Style','pushbutton', ...
'Units','normalized', ...
'Position',[izq posY anchBot altBot], ...
'String',labelStr,'callback',callbackStr);

labelStr='Aplicar';
callbackStr='dct03("aplicar")';
posY=punSuper-4*(altBot+espacio);
gene=uicontrol( ...
'Style','pushbutton', ...
'Units','normalized', ...
'Position',[izq posY anchBot altBot], ...
'String',labelStr,'callback',callbackStr);

labelStr='Cerrar';
callbackStr='close(gcf)';
posY=punSuper-5*(altBot+espacio);
gene=uicontrol( ...
'Style','pushbutton', ...
'Units','normalized', ...
'Position',[izq posY anchBot altBot], ...
'String',labelStr, ...
'Callback',callbackStr);

altBarra = 0.04; anchobar = anchBot+2*frmBorde;
inferiorBs=frmInferior-frmBorde;
izqBs = izq-frmBorde;
```

```
anchotext = anchobar/2;
posBarra = [izqBs inferiorBs anchobar altBarra];
callbackStr = 'dct03("barrita")';
barra = uicontrol( ...
    'Style','slider', ...
    'Units','normalized', ...
    'Position',posBarra, ...
    'Value',0, ...
    'min',0, ...
    'max',64, ...
    'Interruptible','off', ...
    'Callback',callbackStr);
%
c = get(gcf,'Color');
if [.298936021 .58704307445 .114020904255]*c<.5,
    colorDfon = [1 1 1];
else
    colorDfon = [0 0 0];
end

posRango = [izqBs inferiorBs-altBarra anchotext altBarra];
uicontrol( ...
    'Style','text', ...
    'Units','normalized', ...
    'Position',posRango, ...
    'Horiz','left', ...
    'Background',c, ...
    'Foreground',colorDfon, ...
    'String','0');
posRango = [izqBs+anchobar/2 inferiorBs-altBarra anchotext altBarra];
uicontrol( ...
    'Style','text', ...
    'Units','normalized', ...
    'Position',posRango, ...
    'Horiz','right', ...
    'Background',c, ...
    'Foreground',colorDfon, ...
    'String','64');

valor=get(barra,'value');
val_med=round(valor);
pres=num2str(val_med);
%%

posRango = [izqBs+anchobar/4 inferiorBs-altBarra anchotext altBarra];
men01=uicontrol( ...
    'Style','text', ...
```

```

    'Units','normalized', ...
        'Position',posRango, ...
        'Horiz','center', ...
        'Background',c, ...
    'Foreground',colorDfon, ...
        'String',pres);

aPos = [izqBs inferiorBs+2*altBarra anchobar anchobar];
ma8 = axes( ...
    'Units','normalized', ...
    'Position',aPos, ...
    'XTick',[],'YTick',[], ...
    'ydir','reverse', ...
    'Box','on', ...
    'xlim',[1 9], ...
    'ylim',[1 9], ...
    'clim',[0 1]);
title('Coeficientes DCT')
axes(ma8),cdsup=surface(ones(9),'facecolor','flat','edgecolor',[1 1 1]/2);

%
bordeGen=0.02;
altCuad=0.32;
ancCuad=0.32;
%
espLibhor=1-2*bordeGen-2*altCuad;
guardaHor=espLibhor/4;
%%
%%
espLibver=izq-bordeGen-2*ancCuad;
guardaVer=espLibver/3;
posIzq=bordeGen+guardaVer;

%%
altdejeInf=bordeGen+guardaHor;
cimA= axes(...
    'Units','normalized', ...
    'Position',[posIzq altdejeInf ancCuad altCuad], ...
    'XTick',[],'YTick',[], ...
    'Box','on');

altdejeSup= 1-bordeGen-altCuad-guardaHor;

cimR= axes(...
    'Units','normalized', ...
    'Position',[posIzq altdejeSup ancCuad altCuad], ...
    'XTick',[],'YTick',[], ...

```

```

'Box','on');

%%
disIzqdos=izq-guardaVer-ancCuad;

%%
entra='0';
cimMen= axes(...
    'Units','normalized', ...
    'Position',[disIzqdos altdejeInf ancCuad altCuad], ...
    'XTick',[0.6],'YTick',[],...
    'xtickLabel',[],...
    'GridLineStyle','-','...',
    'xGrid','on',...
    'Box','on','color',[0.1 0.2 0.3]); % aqui 0.85 0.9 0.9
    posIzqtex=0.0; %0.02
limSuptex=0.0; %0.05
limInfTex=0.0; %0.05
espaTex=1-limSuptex-limInfTex;
cd_men=struct('men1',{'Tamaño original','Tamaño comprimido',...
    'Factor de compresión',...
    'MSE','PSNR'});
[rdt,cdt]=size(cd_men);
pasoTex=espaTex/cdt;
yTex=1-limSuptex;

for col=1:cdt
    yTex=yTex-pasoTex;
    vec_pt(col,1)=yTex;
    text(posIzqtex ,yTex,cd_men(1,col).men1, ...
        'HorizontalAlignment','Left', ...
        'VerticalAlignment','Bottom',...
        'Color','k', ...
        'FontWeight','light', ...
        'FontSize',8);
end

%
coldato=0.65;%%
meR1=text(coldato,vec_pt(1,1),entra, ...
    'HorizontalAlignment','Left', ...
    'VerticalAlignment','Bottom',...
    'Color','k', ...
    'FontWeight','light', ...
    'FontSize',8);

meR2=text(coldato,vec_pt(2,1),entra, ...

```

```
'HorizontalAlignment','Left', ...
'VerticalAlignment','Bottom',...
'Color','k', ...
'FontWeight','light', ...
'FontSize',8);

meR3=text(coldato,vec_pt(3,1),entra, ...
'HorizontalAlignment','Left', ...
'VerticalAlignment','Bottom',...
'Color','k', ...
'FontWeight','light', ...
'FontSize',8);

meR4=text(coldato,vec_pt(4,1),entra, ...
'HorizontalAlignment','Left', ...
'VerticalAlignment','Bottom',...
'Color','k', ...
'FontWeight','light', ...
'FontSize',8);

meR5=text(coldato,vec_pt(5,1),entra, ...
'HorizontalAlignment','Left', ...
'VerticalAlignment','Bottom',...
'Color','k', ...
'FontWeight','light', ...
'FontSize',8);

%

cimErr= axes(...
'Units','normalized', ...
'Position',[disIzqdos altdejeSup ancCuad altCuad], ...
'XTick',[],'YTick',[], ...
'Box','on');
%%
listMane = [cimA cimR ma8 barra men01 cimMen cimErr ...
meR1 meR2 meR3 meR4 meR5];
set(man_fig, ...
'UserData',listMane);

leima01('lena256','bmp');
return
end

if strcmp(accion,'aplicar')
set(gcf,'Pointer','watch');
```

```
load respa01;
%%
for coo=1:C
    for ree=1:R
        pue_02=mat_trans(:,:,ree,coo);
        mat_cero(:,:,ree,coo)=maconce8(pue_02,val_med);
    end
end
%

%%

[matbid,rrr,ccc]=cdcuaados(mat_cero,val_med);
%codifi_01(matbid,rrr,ccc);%%
%codifi_02(matbid,rrr,ccc);%%
%codifi_03(matbid,rrr,ccc);
codifi_04(matbid,rrr,ccc);%%

%%

%%
%[matd2d,rblo,cblo]=decodifi_01;%%
%[matd2d,rblo,cblo]=decodifi_02;%%
%[matd2d,rblo,cblo]=decodifi_03;%%
[matd2d,rblo,cblo]=decodifi_04;%%

mcero_deco=cddosacua(matd2d,rblo,cblo);
%
%%.
normaliza;%%
for co=1:cblo
    for re=1:rblo
        pue_01=mcero_deco(:,:,re,co);
        mcero_deco(:,:,re,co)=round(pue_01.*mjpeg);
    end
end
%

%
for cco=1:cblo
    for rre=1:rblo
        pue_03=mcero_deco(:,:,rre,cco);
        inuedct=idct(idct(pue_03).)';
        mat_invtra(:,:,rre,cco)=round(inuedct);%si se usa inundct8 no se debe usar round
    end
end
%
```

```
mat_invtra=mat_invtra+128;%%  
  
%%  
mat_sal=individe8(mat_invtra,rblo,cblo);  
%%  
  
%%  
%%  
sal_act=zeros(re,ce);  
[rs,cs]=size(mat_sal);  
for y=1:cs  
    for z=1:rs  
        sal_act(z,y)=mat_sal(z,y);  
    end  
end  
%  
  
%%  
mLocal=get(gcf,'Userdata');  
colormap(mapa);  
axes(mLocal(1)),image(uint8(sal_act)), title('Imagen Reconstruida');%cuidado con el  
set(mLocal(1),'Xtick',[],'Ytick',[]); 128  
%calderro(imdoble,sal_act);%%  
%calderro_02(imdoble,sal_act);%%  
%calderro_03(imdoble,sal_act);%%  
calderro_04(imdoble,sal_act);%%  
  
set(gcf,'Pointer','arrow');  
return  
end  
  
if strcmp(accion,'barrita')  
    load respa01;  
    mane=get(gcf,'userdata');  
    valor=get(mane(4),'value');  
    val_med=round(valor);  
    pres=num2str(val_med);  
    set(mane(5),'string',' ');  
    set(mane(5),'string',pres);  
    mat_dcero=zeros(9);  
    masca8=maconce8(ones(8),val_med);  
    for rm=1:8  
        for cm=1:8  
            mat_dcero(rm,cm)=masca8(rm,cm);  
        end  
    end  
end
```

```
axes(man3),cla,surface(mat_dcero);
save respa01 mapa R C mat_trans val_med totpixPer re ce imdoble
return
end
if strcmp(accion,'lena256')
    leima01('lena256','bmp')
    return
elseif strcmp(accion,'Matrix')
    leima01('Matrix','bmp')
    return
elseif strcmp(accion,'camera')
    leima01('camera','bmp')
    return
end
```