**clc**

**clear all;% clear all variables from previous sessions**

**close all;**

**covername = input('Enter image file name with extension: ', 's');**

**messagename = input('Enter message image file name with extension: ', 's');**

**cover = imread(covername);**

**sz = size(cover);**

**rows = sz(1,1);**

**cols = sz(1,2);**

**colors = max(max(cover));**

**%r=1;**

**%for i=1:rows**

**% for j=1:3:cols**

**% rgb(r,1)=cover(i,j);**

**% rgb(r,2)=cover(i,j+1);**

**% rgb(r,3)=cover(i,j+2);**

**% r=r+1;**

**% end**

**%end**

**%gray=rgb2gray(rgb);**

**%cover=gray;**

**fd = fopen (messagename, 'r');**

**message = fgetl(fd);**

**messagelength = length(message);**

**figure(1), imshow(cover); title('Original Image (Cover Image)');**

**%disp(message);**

**%cover=double(cover);**

**%message=double(message);**

**message = uint8(message);**

**coverzero = cover;**

**%disp(coverzero);**

**%coverzero=imread('GrayScale.bmp');**

**quant\_multiple = 1;**

**blocksize = 8;**

**DCT\_quantizer = ...**

**[ 16 11 10 16 24 40 51 61; ...**

**12 12 14 19 26 58 60 55; ...**

**14 13 16 24 40 57 69 56; ...**

**14 17 22 29 51 87 80 62; ...**

**18 22 37 56 68 109 103 77; ...**

**24 35 55 64 81 104 113 92; ...**

**49 64 78 87 103 121 120 101; ...**

**72 92 95 98 112 100 103 99 ];**

**%figure(1)**

**%image(coverzero)**

**figure(2);imshow(coverzero);**

**%colormap(map)**

**title('Original image');**

**%figure(2)**

**%coverzero = coverzero - ceil(colors/2);**

**%figure(2);imshow(coverzero);**

**pad\_cols = (1 - (cols/blocksize - floor(cols/blocksize))) \* blocksize;**

**if pad\_cols == blocksize, pad\_cols = 0; end**

**pad\_rows = (1 - (rows/blocksize - floor(rows/blocksize))) \* blocksize;**

**if pad\_rows == blocksize, pad\_rows = 0; end**

**for extra\_cols = 1:pad\_cols**

**coverzero(1:rows, cols+extra\_cols) = coverzero(1:rows, cols);**

**end**

**cols = cols + pad\_cols; % coverzero is now pad\_cols wider**

**for extra\_rows = 1:pad\_rows**

**coverzero(rows+extra\_rows, 1:cols) = coverzero(rows, 1:cols);**

**end**

**rows = rows + pad\_rows; % coverzero is now pad\_rows taller**

**for row = 1: blocksize: rows**

**for col = 1: blocksize: cols**

**DCT\_matrix = coverzero(row: row + blocksize-1, col: col + blocksize-1);**

**DCT\_matrix = DCT2(DCT\_matrix);**

**% quantize it (levels stored in DCT\_quantizer matrix):**

**%DCT\_matrix = floor (DCT\_matrix ...**

**% ./ (DCT\_quantizer(1:blocksize, 1:blocksize) \* quant\_multiple) + 0.5);**

**DCT\_matrix = round(DCT\_matrix ...**

**./ (DCT\_quantizer(1:blocksize, 1:blocksize) \* quant\_multiple));**

**%DCT\_matrix=round(DCT\_matrix);**

**% place it into the compressed-image matrix:**

**jpeg\_img(row: row + blocksize-1, col: col + blocksize-1) = DCT\_matrix;**

**end**

**end**

**figure(3);hist(jpeg\_img);**

**figure(4);imshow(jpeg\_img);**

**bitlength=1;**

**%messagebit=zeros(messagelength\*8);**

**for i=1:messagelength**

**%imbed=7;**

**for imbed=1:8**

**messageshift=bitshift(message(i),8-imbed);**

**showmess=uint8(messageshift);**

**showmess=bitshift(showmess,-7);**

**messagebit(bitlength)=showmess;**

**bitlength=bitlength+1;**

**%coverindex = coverindex+1;**

**end**

**end**

**%embedding**

**i=1;**

**for row=1:rows**

**for col=1:cols**

**x=jpeg\_img(row,col);**

**if (x~=0) && (x~=1)**

**r=mod(x,2);**

**if r==0 %**

**if messagebit(i)==1**

**x=x+1;**

**end**

**else**

**if messagebit(i)==0**

**x=x-1;**

**end**

**end**

**i=i+1;**

**end**

**jpeg\_img(row,col)=x;**

**if i==bitlength**

**break;**

**end**

**end**

**if i==bitlength**

**break;**

**end**

**end**

**figure(5);hist(jpeg\_img);**

**% Reconstructing image**

**recon\_img = coverzero - coverzero; % zero the matrix for the reconstructed image**

**for row = 1: blocksize: rows**

**for col = 1: blocksize: cols**

**IDCT\_matrix = jpeg\_img(row: row + blocksize-1, col: col + blocksize-1);**

**%IDCT\_matrix = floor(idct2(IDCT\_matrix .\* ((DCT\_quantizer(1:blocksize, 1:blocksize) \* quant\_multiple))-0.5));**

**%IDCT\_matrix = floor(idct2(IDCT\_matrix .\* (DCT\_quantizer(1:blocksize, 1:blocksize) \* quant\_multiple)));**

**IDCT\_matrix = round(idct2(IDCT\_matrix .\* (DCT\_quantizer(1:blocksize, 1:blocksize) \* quant\_multiple)));**

**recon\_img(row: row + blocksize-1, col: col + blocksize-1) = IDCT\_matrix;**

**end**

**end**

**%recon\_img = recon\_img + ceil(colors/2);**

**%coverzero = coverzero + ceil(colors/2);**

**% Clip off padded rows and columns**

**rows = rows - pad\_rows;**

**cols = cols - pad\_cols;**

**recon\_img = recon\_img(1:rows, 1:cols);**

**figure(6);imshow(recon\_img);**

**%disp(recon\_img);**

**%recon\_img = recon\_img - ceil(colors/2);**

**pad\_cols = (1 - (cols/blocksize - floor(cols/blocksize))) \* blocksize;**

**if pad\_cols == blocksize, pad\_cols = 0; end**

**pad\_rows = (1 - (rows/blocksize - floor(rows/blocksize))) \* blocksize;**

**if pad\_rows == blocksize, pad\_rows = 0; end**

**for extra\_cols = 1:pad\_cols**

**recon\_img(1:rows, cols+extra\_cols) = recon\_img(1:rows, cols);**

**end**

**cols = cols + pad\_cols; % coverzero is now pad\_cols wider**

**for extra\_rows = 1:pad\_rows**

**recon\_img(rows+extra\_rows, 1:cols) = recon\_img(rows, 1:cols);**

**end**

**rows = rows + pad\_rows; % coverzero is now pad\_rows taller**

**jpeg\_img=jpeg\_img-jpeg\_img;**

**for row = 1: blocksize: rows**

**for col = 1: blocksize: cols**

**DCT\_matrix = recon\_img(row: row + blocksize-1, col: col + blocksize-1);**

**DCT\_matrix = DCT2(DCT\_matrix);**

**% quantize it (levels stored in DCT\_quantizer matrix):**

**%DCT\_matrix = floor (DCT\_matrix ...**

**% ./ (DCT\_quantizer(1:blocksize, 1:blocksize) \* quant\_multiple) + 0.5);**

**DCT\_matrix = round (DCT\_matrix ...**

**./ (DCT\_quantizer(1:blocksize, 1:blocksize) \* quant\_multiple));**

**%DCT\_matrix=round(DCT\_matrix);**

**% place it into the compressed-image matrix:**

**jpeg\_img(row: row + blocksize-1, col: col + blocksize-1) = DCT\_matrix;**

**%disp(jpeg\_img(row: row + blocksize-1, col: col + blocksize-1));**

**end**

**end**

**stego=jpeg\_img;**

**%stego = uint8(jpeg\_img);**

**%stego = int8(jpeg\_img);**

**%disp(stego(1:8,1:8));**

**%stego = uint8(jpeg\_img);**

**%disp(stego(1:8,1:8));**

**stegoindex=1;**

**imbed=1;**

**messagechar=0;**

**messageindex=1;**

**%for i=1:(messagelength\*8)**

**for row=1:rows**

**for col=1:cols**

**stegomessage = stego(row,col);**

**if (stegomessage~=0)&&(stegomessage~=1)**

**r=mod(stegomessage,2);**

**if (r==0)**

**showmess=0;**

**else showmess=1;**

**end**

**showmess=uint8(showmess);**

**%showmess=bitshift(stegomessage,7);**

**showmess=bitshift(showmess,(imbed-1));**

**messagechar=uint8(messagechar+showmess);**

**stegoindex = stegoindex+1;**

**imbed=imbed+1;**

**if (imbed==9)**

**messagestring(messageindex)=messagechar;**

**messageindex=messageindex+1;**

**messagechar=0;**

**imbed=1;**

**end**

**end**

**if (stegoindex==messagelength\*8)**

**break;**

**end**

**end**

**if (stegoindex==messagelength\*8)**

**break;**

**end**

**end**

**%end**

**disp(messagestring);**