

MATLAB

Project Report

Data Hiding using Image Encryption in MATLAB

GROUP MEMBERS:

Suvimal Yashraj 140911394 Roll no 48

Trishati Rao 140905013 Roll no 4

Riju Khatri 140911088 Roll no 43

**Synopsis:**

In our project we have used methods to hide text and images in an image which include, encryption of the image, embedding of data and extraction of data. This consists of two phases, the encryption phase and the decryption phase. Different techniques have been used for encrypting text and image respectively.

The objective of this project is to provide an efficient data hiding technique with minimal distortion of the encrypting image using Image Encryption in which data and image can be retrieved independently.

Encrypting Text

The first part of the project is hiding text using Image Encryption .Here, the text to be hidden is written in a text file. The maximum number of characters that can be hidden in the image is equal to the product of width and height of the image, in pixels. The image in first converted into binary values and stored into a matrix .Then the text in the file is also converted to binary values and stored in a matrix. After that the corresponding binary values of the two matrices are added, and this is how data encryption has been performed.

Decrypting Text

The second part of the project is decrypting the text hidden in an image. For decryption, the encrypted image and the original image are compared, and the corresponding binary values are stored in a matrix, and then converted to text and written in a another file.

Encrypting an image

The third part is encrypting an image into another image. This can be done by increasing the size of image 1(image used to hide image 2) then encoding pixel details of image 2 into image 1. By doing this the quality of image 2 will not be affected. Also only 2 bits in every pixel of image 1 is used for encoding this is because the encoded image does not show any patches of image 2. The bits in every pixel of the image2 is split into four and placed in four different locations in image1. So it provides some sort of encryption. So it will be hard for others to decode the hidden image.

Decrypting the image

For decrypting the image, the encrypted image’s resolution is decreased and made the same as the original image’s resolution. Then the matrix that had been formed while encrypting is used as a key to decrypt the image.

**Source Code:**

%Program for hiding an image inside the other

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% ---Only two bit in pixel of the image 1 is affected

% ---image2 is split & store in two diagonally opposite quadrants

% Algorithm

% the first image is resize to double its original size

% logically the image is divided to four partitions

% the bits of the image to be hid is stored in the first image as

% shown below

% |-------------|

% |d7,d6 |d3,d2 |

% |-------------|

% |d1,d0 |d5,d4 |

% |\_\_\_\_\_\_\_\_\_\_\_\_ |

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*program\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

clc; % clear the command window

clear; % clear the workspace

disp(' ');

disp(' \*\*\*\*\* IMAGE HIDER 2.0 \*\*\*\*\*');

disp('\_\_\_Program for hidimg one image inside the other image\_\_\_');

disp(' ');

disp('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

task = input('---Encode Text into Image :- 1 \n---Decode Text from Image :- 2\n---Encrypt image into another image :-3\n---Decrypt image from the encrypted image :-4\n Enter your task:');

% select task

if isempty(task)

task=1;

end

if task == 1

FID = fopen('myfile.txt', 'rb'); %opening text file, integer file identifier obtained in fid

Str = fread(FID, [1,inf], 'char');

%reading text file, reads binary data from the specified file and writes it into matrix Str

%size[1,inf] - read elements to fill an 1-by-inf matrix, in column order inf read to the end of the file.

%reads the file according to the data format specified

fclose(FID); %closing the file

Str=uint16(Str); %converting to unsigned 16 bit integer for proper calculation.

%Any element outside limit gets rounded off to nearest endpoint

x=imread('original.png'); %reading the image file into x matrix which contains binary values

x=uint16(x); %conversion to 16 bit

[x\_row,x\_col]=size(x); %returns the dimensions of the matrix x

c=numel(Str); %counting characters (numel=number of elements)

a=1;

%encrypting loop

for i=1:x\_row

for j=1:x\_col

if(a<=c)

if(x(i,j)+Str(a)>255) %if greater than 255 ie 8 bits then putting it in 8 bit form.

temp=x(i,j)+Str(a)-256;

else

temp=x(i,j)+Str(a);

end

z(i,j)=uint8(temp); %converting back to default

else

z(i,j)=uint8(x(i,j)); %putting original image bits for return of encrypted image

end

a=a+1; %incrementing count of characters in text file.

end

end

imwrite(z,'encrypted.png') %writing the encrypted data as pixels in image

imshow(z)

end

if task ==2

x=imread('encrypted.png'); %reading encrypted image

y=imread('original.png'); %reading non-encrypted image

x=uint16(x); %16 bit conversion

y=uint16(y); %16 bit conversion

[x\_row, x\_col]=size(x);

b=0;k=1;

%decrypting loop

for i=1:x\_row

for j=1:x\_col

if(x(i,j)>=y(i,j))

a=x(i,j)-y(i,j);

else

a=256+x(i,j)-y(i,j);

end

if(a~=0)

z(k)=uint8(a);

k=k+1;

else

b=1;

break;

end

end

if(b==1)

break;

end

end

fid=fopen('decrypted.txt','w'); %creating text file to write decrypted data

for i=1:k-1

fprintf(fid,'%c',z(i)); %writing to file

end

disp('The image has been decrypted and the text in written in decrypt.txt')

end

if task == 3

% reads two image files

x = imread(input(' Welcome to Encoder\n Enter the first image file name: ','s'));

y = imread(input(' Enter the image to be encrypted : ','s'));

% check compatibility

sx = size(x);

sy = size(y);

x=imresize(x,[2\*sy(1),2\*sy(2)]); %x is 2 times the size of y

% clearing Ist files last two lsb bits & moving IInd files msb bits to lsb bits

x1 = bitand(x,252); %bitwise 'and' clearing the LSB bits

y1 = bitshift(y,-4); %shifting to the right, or dividing by 2^ABS(4) and truncating to an integer y1

% we get 4 bits of the msb of image 2

y1\_= bitand(y1,12); %and with 1100 so we get d7 and d6

y1\_= bitshift(y1\_,-2); %2 MSB1 bits d7 and d6

y1 = bitand(y1,3); % and with 0011 so we get MSB2 d5 and d4

% clearing II image's msb bits

y\_lsb1 = bitshift(bitand(y,12),-2);

y\_lsb2 = bitand(y,3);

% inserting IInd to Ist file

z=x1;

for j=1:sy(2) % y variation

for i=1:sy(1) % x variation

for k=1:3

%we enter the bits to each quadrant for further encryption.

% IInd quadrant

z(i,j,k) = bitor(x1(i,j,k), y1\_(i,j,k));

% IV th quadrant

z(i+sy(1),j+sy(2),k) = bitor(x1(i+sy(1),j+sy(2),k), y1(i,j,k));

% I st quadrant

z(i+sy(1),j,k) = bitor(x1(i+sy(1),j,k), y\_lsb1(i,j,k));

% IIIrd quadrant

z(i,j+sy(2),k) = bitor(x1(i,j+sy(2),k), y\_lsb2(i,j,k));

end

end

end

% display the first image

figure(1)

image(x);

xlabel(' Ist Image ');

% display IInd image

figure(2);

image(y);

xlabel(' IInd Image ');

% display encoded image

figure(3);

image(z);

xlabel(' Encoded Image ');

% saving file

sav=input('Do you want to save the file y/n [y] ','s');

if isempty(sav)

sav='y';

end

if sav == 'y'

name=input('Enter a name for the encoded image: ','s');

if isempty(sav)

name='encoded\_temp';

end

name=[name,'.bmp']; % concatination

imwrite(z,name,'bmp');

end

if task==4

% Decoding encoded image

clear;

z=imread(input(' Welcome to Decoder\n Enter the image file to be decoded:','s'));

sy = size(z)/2; % take the size of input file

% xo is fist file- obtained by clearing lsb bits, yo is IInd file right

% shifting z by 4 bits

xo=bitand(z,252);

xo=imresize(xo,[sy(1),sy(2)]); % reduce the resolution to half so

%that it becoms the original image's resolution

for j=1:sy(2) % y variation

for i=1:sy(1) % x variation

for k=1:3

zout1(i,j,k) = bitshift(bitand(z(i,j,k),3),2);

zout2(i,j,k) = bitand(z(i+sy(1),j+sy(2),k), 3);

zout3(i,j,k) = bitshift(bitand(z(i+sy(1),j,k),3),2);

zout4(i,j,k) = bitand(z(i,j+sy(2),k),3);

end

end

end

zout = bitshift((zout1+zout2),4)+zout3+zout4;

yo = zout;

% display Ist & IInd image from encoded image

figure(4);

image(xo);

xlabel('Ist Decoded Image ');

figure(5);

image(yo);

xlabel('IInd Decoded Image');

% saving file

sav=input('Do you want to save the file y/n [y] ','s');

if isempty(sav)

sav='y';

end

if sav == 'y'

name1=input('Enter a name for the first image: ','s');

name2=input('Enter a name for the second image: ','s');

if isempty(name1)

name1 = 'Ist\_temp';

end

if isempty(name2)

name2 = 'IInd\_temp';

end

name1 = [name1,'.bmp'];

name2 = [name2,'.bmp'];

imwrite(xo,name1,'bmp');

imwrite(yo,name2,'bmp');

end

end

end

**References**

<http://www.arpnjournals.com/jeas/research_papers/rp_2014/jeas_0514_1083.pdf>

http://www.ijircce.com/upload/2014/march/32\_Inverse.pdf