ABSTRACT

In this project we have used **DISCRETE COSINE TRANSFORM** algorithm using MATLAB, which compresses the image with a good compression ratio. Cameras are nowadays being provided with more and more megapixels to improve the quality of captured images. With improvement in image quality, size of the image file also increases.

Due to speed limitation of the Internet, it takes more time to upload good-quality images that are of bigger sizes. A user needs to compress the image without degrading its quality. Mobile manufacturers need algorithms in their cameras that enable storing the images in reduced sizes without degrading their quality.

The image is read through MATLAB to capture its pixels. After obtaining the compressed image, peak-signal-noise ratio (PSNR) and mean-square error (MSE) are calculated using the following relationships:

$$MSE = \frac{\sum_{M,N}(Image1(m,n) - Image2(m,n))^2}{mxn}$$

where m and n are the number of rows and columns. Image1 and Image2 are the original and compressed images, respectively.

After compression, there should not be much change in the quality of the image. MSE indicates an error between the original image and compressed image. It should be as small as possible.

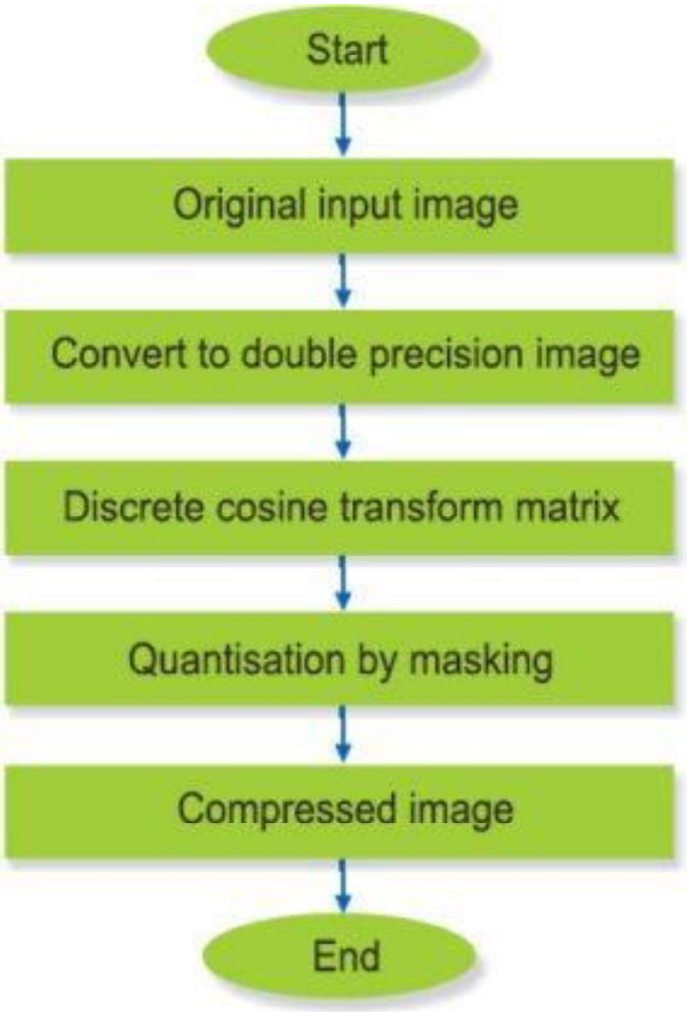
where R is the maximum fluctuation in the input image data type (maximum possible pixel value of image). PSNR is related to MSE and it gives the amount of noise in a compressed image. PSNR should be as high as possible.

$$PSNR = 10 \log_{10} \left(\frac{R^2}{MSE} \right)$$

All equations are implemented in MATLAB in the form of functions.

COMPRESSION PROCESS:

The flowchart shows the compression process



WORKING OF CODE:

VARARGOUT is an output variable in a function definition statement that enables the function to return any number of output arguments. Specify varargout using lowercase characters, and include it as the last output argument after any explicitly declared outputs.

When the function executes, varargout is a 1-by-N cell array, where N is the number of outputs requested after the explicitly declared outputs.

VARARGIN is an input variable in a function definition statement that enables the function to accept any number of input arguments. Specify varargin using lowercase characters, and include it as the last input argument after any explicitly declared inputs.

When the function executes, varargin is a 1-by-N cell array, where N is the number of inputs that the function receives after the explicitly declared inputs. However, if the function receives no inputs after the explicitly declared inputs, then varargin is an empty cell array.

VARARGOUT is an output variable in a function definition statement that enables the function to return any number of output arguments. Specify varargout using lowercase characters, and include it as the last output argument after any explicitly declared outputs.

When the function executes, varargout is a 1-by-N cell array, where N is the number of outputs requested after the explicitly declared outputs.

IMAGECOMPRESSION1, by itself, creates a new IMAGECOMPRESSION1 or raises the existing singleton.

H = IMAGECOMPRESSION1 returns the handle to a new IMAGECOMPRESSION1 or the handle to the existing singleton.

IMAGECOMPRESSION1('CALLBACK',hObject,eventData,handles,...) calls the local function named CALLBACK in IMAGECOMPRESSION1.M with the given input arguments.

IMAGECOMPRESSION1('Property','Value',...) creates a new IMAGECOMPRESSION1 or raises the existing singleton*. Starting from the left, property value pairs are applied to the GUI before ImageCompression1_OpeningFcn gets called. An unrecognized property name or invalid value makes property application stop. All inputs are passed to ImageCompression1_OpeningFcn via varargin.

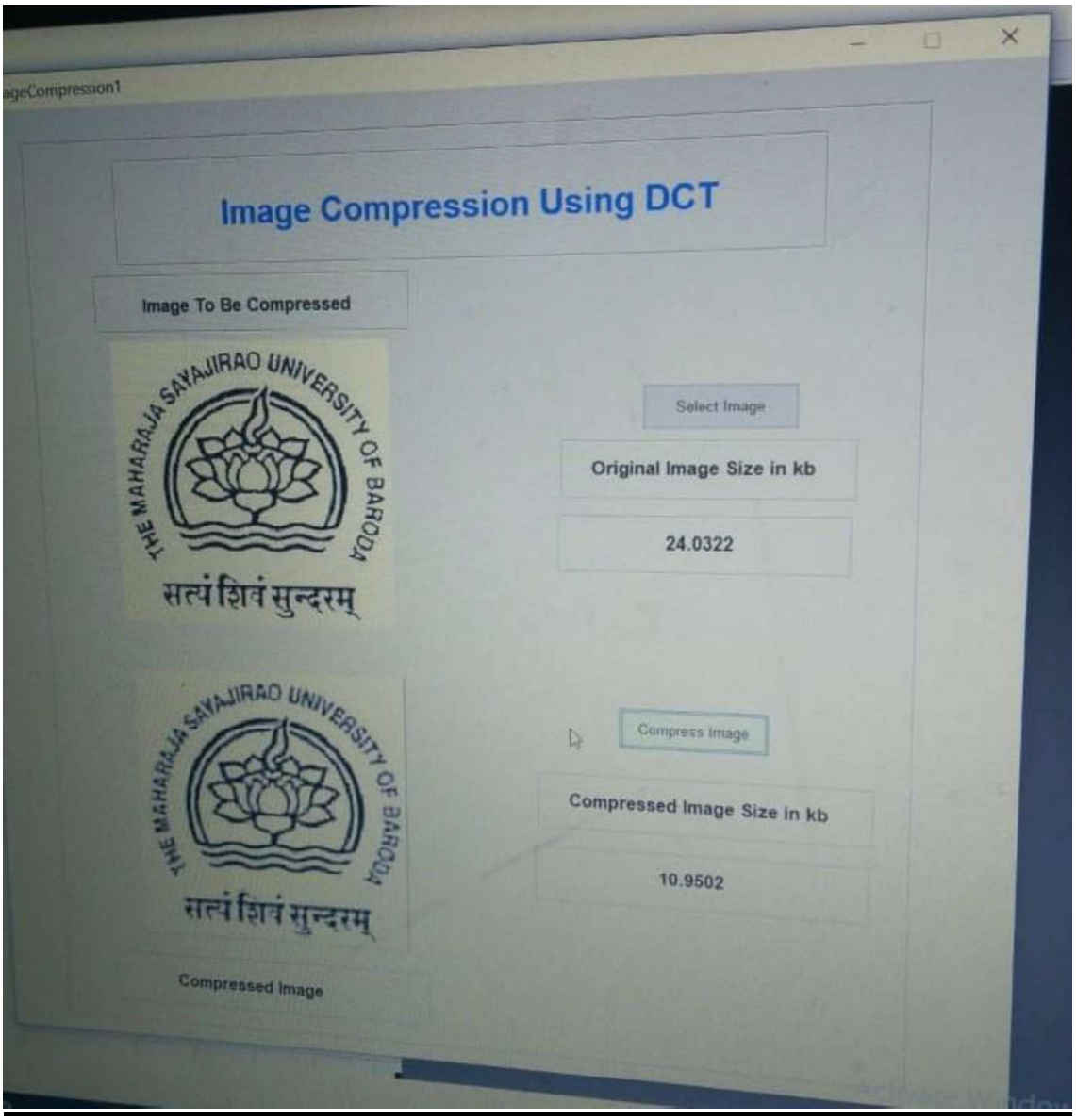
MATLAB CODE:

```
function varargout = ImageCompression1(varargin)


gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @ImageCompression1_OpeningFcn, ...
                   'gui_OutputFcn',  @ImageCompression1_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before ImageCompression1 is made visible.
function ImageCompression1_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to ImageCompression1 (see VARARGIN)

% Choose default command line output for ImageCompression1
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);
guidata(hObject, handles);
set(handles.axes1,'visible','off')
set(handles.axes2,'visible','off')
axis off
axis off
% UIWAIT makes ImageCompression1 wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = ImageCompression1_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
```

```
% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global file_name;
%guidata(hObject,handles)
file_name=uigetfile({'*.bmp;*.jpg;*.png;*.tiff;';'*.*'},'Select an Image
File');
fileinfo = dir(file_name);
SIZE = fileinfo.bytes;
Size = SIZE/1024;
set(handles.text7,'string',Size);
imshow(file_name,'Parent', handles.axes1)

% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global file_name;
if(~ischar(file_name))
   errordlg('Please select Images first');
else
    I1 = imread(file_name);
% I1 = imread('chicken.jpg');
I = I1(:,:,1);
I = im2double(I);
T = dctmtx(8);
B = blkproc(I,[8 8],'P1*x*P2',T,T');
mask = [1   1   1   1   0   0   0   0
        1   1   1   0   0   0   0   0
        1   1   0   0   0   0   0   0
        1   0   0   0   0   0   0   0
        0   0   0   0   0   0   0   0
        0   0   0   0   0   0   0   0
        0   0   0   0   0   0   0   0
        0   0   0   0   0   0   0   0];
B2 = blkproc(B,[8 8],'P1.*x',mask);
I2 = blkproc(B2,[8 8],'P1*x*P2',T',T);

I = I1(:,:,2);
I = im2double(I);
T = dctmtx(8);
B = blkproc(I,[8 8],'P1*x*P2',T,T');
mask = [1   1   1   1   0   0   0   0
        1   1   1   0   0   0   0   0
        1   1   0   0   0   0   0   0
        1   0   0   0   0   0   0   0
        0   0   0   0   0   0   0   0
        0   0   0   0   0   0   0   0
        0   0   0   0   0   0   0   0
        0   0   0   0   0   0   0   0];
B2 = blkproc(B,[8 8],'P1.*x',mask);
I3 = blkproc(B2,[8 8],'P1*x*P2',T',T);
```

```
I = I1(:,:,3);
I = im2double(I);
T = dctmtx(8);
B = blkproc(I,[8 8],'P1*x*P2',T,T');
mask = [1   1   1   1   0   0   0   0
        1   1   1   0   0   0   0   0
        1   1   0   0   0   0   0   0
        1   0   0   0   0   0   0   0
        0   0   0   0   0   0   0   0
        0   0   0   0   0   0   0   0
        0   0   0   0   0   0   0   0
        0   0   0   0   0   0   0   0];
B2 = blkproc(B,[8 8],'P1.*x',mask);
I4 = blkproc(B2,[8 8],'P1*x*P2',T',T);


L(:,:,:)=cat(3,I2, I3, I4);
imwrite(L,'CompressedColourImage.jpg');

fileinfo = dir('CompressedColourImage.jpg');
SIZE = fileinfo.bytes;
Size = SIZE/1024;
set(handles.text8,'string',Size);
imshow(L,'Parent', handles.axes2)
end
```

OUTPUT:


ageCompression1
Image Compression Using DCT
Image To Be Compressed
THE MAHARAJA SAYAJIRAO UNIVERSITY OF BARODA
सत्यं शिवं सुन्दरम्
Select Image
Original Image Size in kb
24.0322
Compress Image
Compressed Image Size in kb
10.9502
Compressed Image

Compressed Image Saved in the Directory Folder