Code

```
original=imread('ttest.jpg');

figure
imshow(original);


 original = double(original);
 Image = original-128;

[Image_Height,Image_Width,Number_Of_Colour_Channels] = size(Image);

% check greyscale
if Number_Of_Colour_Channels == 3 % it is colored
  Image=Image(:,:,1);

end
% initaialize the size &height & Width
Block_Size = 8;
Number_Of_Blocks_Vertically = Image_Height/Block_Size;
Number_Of_Blocks_Horizontally = Image_Width/Block_Size;
Image_Blocks = struct('Blocks',[]);
collect= struct('collect',[]);

 % divide to Blocks
Index = 1;
for Row = 1: +Block_Size:Image_Height
    for Column = 1: +Block_Size: Image_Width

    Row_End = Row + Block_Size - 1;
    Column_End = Column + Block_Size - 1;
    Temporary_Tile = Image(Row:Row_End,Column:Column_End,:);

    %Storing blocks/tiles in structure for later use%
    Image_Blocks(Index).Blocks = Temporary_Tile;
    Index = Index + 1;

    end
end
Q = [
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 81 194 113 92 ;
49 64 78 87 103 121 120 101;
72 92 95 98 121 100 103 99
];
r=10;
C8 = dctmtx(8);
T = r * Q ;
Index=1;
for Row = 1: +Block_Size: Image_Height
    for Column = 1: +Block_Size: Image_Width
```

```
    Row_End = Row + Block_Size - 1;
    Column_End = Column + Block_Size - 1;

 % encoding

    block = Image(Row:Row_End,Column:Column_End,:);
    encoded_block = C8*double(block)*transpose(C8);
        Y = round(encoded_block./T);

    % decoding

    decoded_block = Y.*T ;

    origenal_matrix=transpose(C8)*decoded_block*C8;
     origenal_matrix=origenal_matrix+128;
     origenal_matrix=uint8(origenal_matrix);

    %Storing blocks/tiles in structure for later use%
    Image_Blocks(Index).Blocks = origenal_matrix;
    Index=Index+1;

    end
end
Index=1;
i=0;
% merge
for Row=2:Number_Of_Blocks_Vertically+1
     n= Image_Blocks(1+i).Blocks;
     for Column = 2+i: Number_Of_Blocks_Horizontally+i

     n=[n Image_Blocks(Column).Blocks];
     end
     collect(Row).collect=n;
     i=i+Number_Of_Blocks_Horizontally;

end
arr=collect(1).collect;
for Row=2:Number_Of_Blocks_Vertically+1
    arr=[arr ;collect(Row).collect];
end
 figure
imshow(arr);
```

Explanation for the code

1. We take the image and check is it grayscale or not if not, we convert it to grayscale
2. Dividing images into blocks each block's size is 8*8
3. We take each block and convert it to DCT by the steps in the PDF

4. IN Encoding Part

Get the Y as it is calculated by this Equation

$$y_{i,j} = \text{round}\left(\frac{x_{i,j}}{T_{i,j}}\right)$$

AS X is the DCT_ input and we can calculate T by this Equation

$$T = r \times DCTQ$$

5. IN Decoding Part

To recover the DCT_input we use this Equation

$$\boldsymbol{x_{i,j} = y_{i,j} \times T_{i,j}}$$

And to get the Inverse DCT we used this Equation

$$A = C_N^T \times \hat{A} \times C_N$$

6. Merge the blocks to return the image

7. **Results**

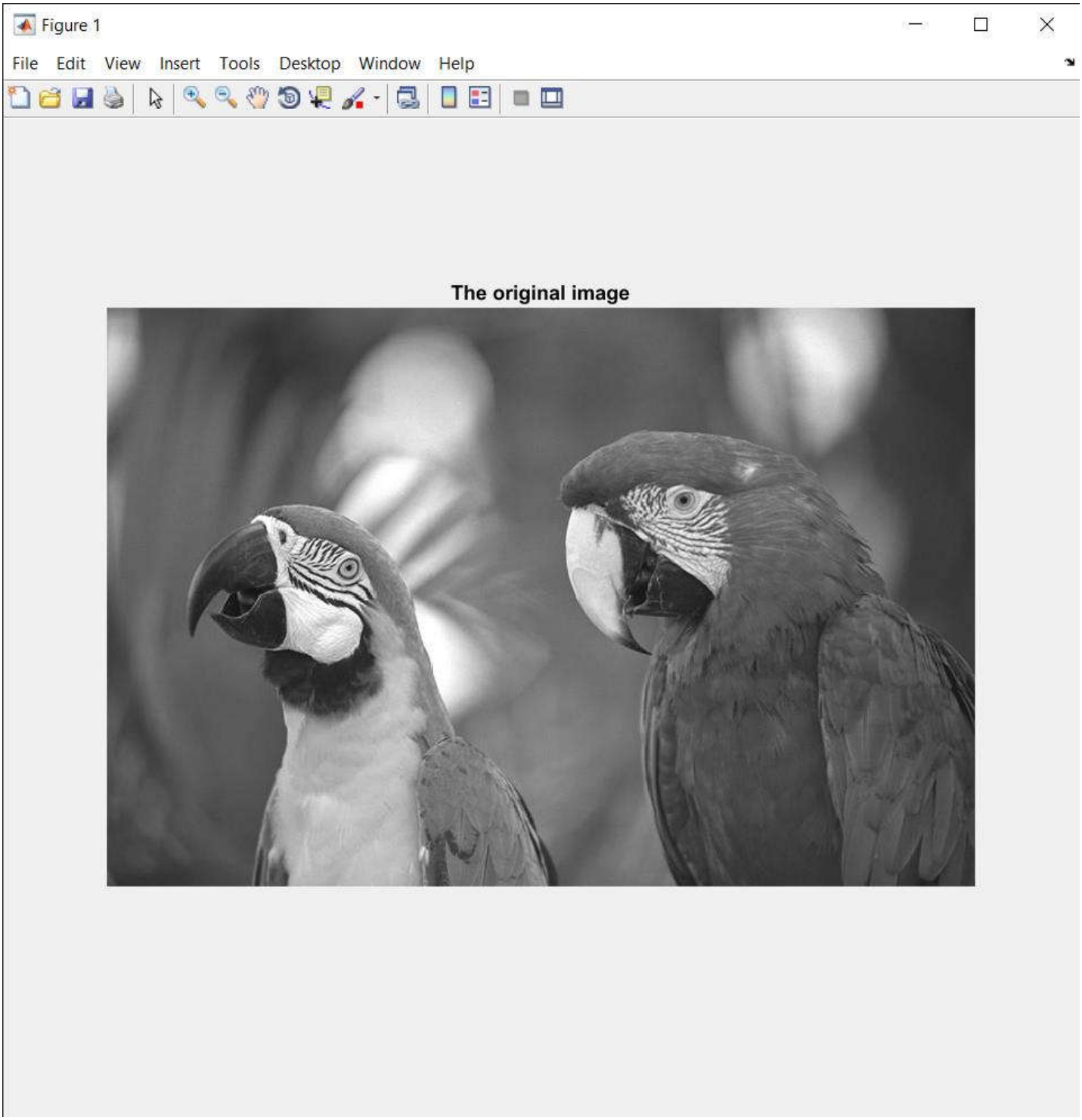
Figure 1
File Edit View Insert Tools Desktop Window Help
The original image

Figure 2

File Edit View Insert Tools Desktop Window Help

Figure 2

File Edit View Insert Tools Desktop Window Help

The image with r=3

Figure 2

File Edit View Insert Tools Desktop Window Help

Figure 2

File Edit View Insert Tools Desktop Window Help