Table of Contents

MOD 5 LAB 3

```
imgOri = imread('Lena128.bmp'); % read in image to imgOri variable
hFig = figure(1); set(hFig, 'position', [100 100 400 350]) % set the
 position and size of our figure

subplot(3,3,1); imshow(imgOri); title('An 8-bit grayscale image'); %
 display the original image in subplot 1
for k = 1 : 8
    v = bitget(imgOri, 9 - k);
    %TODO: extract bitplane into variable
    subplot(3,3, k+1); imshow(v, [0 1]); title('Bitplane #');  %TODO:
 display the extracted bitplane into specified subplot; note that
 subplot(3,3,10-k) allows us to display the bitplanes in descending o
    %TODO: title each subplot to show which bitplane is being
 displayed
end
truesize % adjust display size - call 'help truesize' in the command
 window to learn more.
```

```
Error using imread>get_full_filename (line 566)
File "Lena128.bmp" does not exist.

Error in imread (line 375)
    fullname = get_full_filename(filename);

Error in mod5lab3 (line 2)
imgOri = imread('Lena128.bmp'); % read in image to imgOri variable
```

Q 2

```
img = imread('Lena128.bmp');

qualityFactor_vector = [100:-10:10];
filesize_vector = zeros(size(qualityFactor_vector));

for i = 1:10

    imwrite(img, 'compressedImg.jpg', 'Quality',
 qualityFactor_vector(i))
    fsize = getFilesize('compressedImg.jpg');
    filesize_vector (1, i) = fsize;

    if qualityFactor_vector(i) == 100
        figure(2); subplot(1,4,1); imshow(img); title('100%');
    end
     if qualityFactor_vector(i) == 90
        subplot(1,4,2); imshow(img); title('90%');
     end
     if qualityFactor_vector(i) == 50
        subplot(1,4,3); imshow(img); title('50%');
     end
     if qualityFactor_vector(i) == 10
        subplot(1,4,4); imshow(img); title('10%');
    end
end
```

Q 3

Q 4

```
figure(3)
stem(qualityFactor_vector, filesize_vector);
xlabel('quality factor (%)'); ylabel('filesize (bytes)');
 title('filesize vs. image quality plot');
```

Q 5

The relationship between quality factor and size is quadraexpontential.

Part 3

Q 6

The optimal bitplanes for image hiding are those that use the least significant bits - the rightmost bits.

Q 7

The optimal bitplanes for image hiding are those at positions 0 1 and 2. The compression levels do not change which bitplanes hide the image better.

Q 8

Since we have only used grayscale images, it is possible that this may affectother kinds of images like RGB or HSV, but for the ones that we have been using this should have no effect. The least significant bits are the same for all images.

Part 4

```
img_boat = imread('boat.tiff');
img_umdMark = rgb2gray(imread('UMD_mark_Athletic.tiff')); figure(108);
subplot(1,2,1); imshow(img_boat); title('host image'); subplot(1,2,2);
 imshow(img_umdMark); title('image to be embedded');
```

Q 8 again

```
umdMark_bit  = bitget(img_umdMark, 8);
img_composite = bitset(img_boat, 2, umdMark_bit);

figure (4);
imshow(img_composite);
```

Q 9

```
hostImg_bit = bitget(img_composite, 2);
```

Q 10

```
figure(5); subplot(1,3,1); imshow(img_boat); title('Original');
subplot(1,3,2); imshow(img_composite); title('Composite');
subplot(1,3,3); imshow(hostImg_bit, [0 1]); title('Extracted
 Bitplane');
```

Q 11

```
umdMark_bit  = bitget(img_umdMark, 8);
umdMark_bit2  = bitget(img_umdMark, 1);
img_composite2 = bitset(img_boat, 1, umdMark_bit);
img_composite3 = bitset(img_boat, 4, umdMark_bit);
img_composite4 = bitset(img_boat, 8, umdMark_bit2);

umdMark_bit  = bitget(img_umdMark, 1);

hostImg_bit2 = bitget(img_composite2, 1);
hostImg_bit3 = bitget(img_composite3, 4);
```

```
hostImg_bit4 = bitget(img_composite4, 8);

figure(6); subplot(3,3,1); imshow(img_boat); title('Original');
subplot(3,3,2); imshow(img_composite2); title('Composite 2');
subplot(3,3,3); imshow(hostImg_bit2, [0 1]); title('Extracted Bitplane
 8');
subplot(3,3,5); imshow(img_composite3); title('Composite 3');
subplot(3,3,6); imshow(hostImg_bit3, [0 1]); title('Extracted Bitplane
 8');
subplot(3,3,8); imshow(img_composite4); title('Composite 4');
subplot(3,3,9); imshow(hostImg_bit4, [0 1]); title('Extracted Bitplane
 1');
```

Part 5

Q 12

```
filesize_vector2 = zeros(size(qualityFactor_vector));
filesize_vector3 = zeros(size(qualityFactor_vector));
for i = 1:10

    imwrite(img_composite, 'compressedImg.jpg', 'Quality',
 qualityFactor_vector(i))
    fsize = getFilesize('compressedImg.jpg');
    filesize_vector2 (1, i) = fsize;
     if qualityFactor_vector(i) == 90
        figure (7); subplot(1,3,1); imshow(img_composite);
 title('90%');
     end
     if qualityFactor_vector(i) == 50
        subplot(1,3,2); imshow(img_composite); title('50%');
     end
     if qualityFactor_vector(i) == 10
        subplot(1,3,3); imshow(img_composite); title('10%');
    end
end
for i = 1:10

    imwrite(img_composite2, 'compressedImg.jpg', 'Quality',
 qualityFactor_vector(i))
    fsize = getFilesize('compressedImg.jpg');
    filesize_vector3 (1, i) = fsize;
     if qualityFactor_vector(i) == 90
        figure (8); subplot(1,3,1); imshow(img_composite2);
 title('90%');
     end
     if qualityFactor_vector(i) == 50
        subplot(1,3,2); imshow(img_composite2); title('50%');
     end
     if qualityFactor_vector(i) == 10
        subplot(1,3,3); imshow(img_composite2); title('10%');
    end
```

```
end
```

Q 13

```
img_host = img_boat;
img_toBeEmbedded = imresize(img_umdMark, 1/2, 'bicubic'); % reduce
 image size by 1/2 in both height and width
newImg = zeros(size(img_boat));
for i = 1:512
    for j = 1: 512
        if i <= 256 && j <= 256
            img_host(i,j) = img_toBeEmbedded(i,j);
        end
          if i <= 256 && j > 256
            img_host(i,j) = img_toBeEmbedded(i,j - 256);
          end
          if i > 256 && j <= 256
            img_host(i,j) = img_toBeEmbedded(i - 256,j);
          end
          if i > 256 && j > 256
            img_host(i,j) = img_toBeEmbedded(i - 256 ,j - 256);
          end
    end
end



bigumdMark_bit  = bitget(img_host, 8);
c = bitset(img_boat, 1, bigumdMark_bit);

figure(10); subplot(3,3,1); imshow(c); title('An 8-bit grayscale
 image'); % display the original image in subplot 1
for k = 1 : 8
    v = bitget(c, 9 - k);
    %TODO: extract bitplane into variable
    subplot(3,3, k+1); imshow(v, [0 1]); title('Bitplane #');  %TODO:
 display the extracted bitplane into specified subplot; note that
 subplot(3,3,10-k) allows us to display the bitplanes in descending o
    %TODO: title each subplot to show which bitplane is being
 displayed
end
truesize % adjust display size - call 'help truesize' in the command
 window to learn more.
```

Published with MATLAB® R2019b