### 1. Extraction Bitplanes from Image and Image Reconstruction from Bitplanes

```
% Loading the image from current directory
current_dir = pwd;
file_name = 'slope.tif';
im = imread(strcat(current_dir,'\slope.tif'));
whos im
% converting image from uint8 to standard double precision
im = double(im);
whos im
% Extracting 8 bitplanes as 8 binary images of the original image
bitPlane_1 = mod(bitshift(im, -0), 2);
bitPlane_2 = mod(bitshift(im, -1), 2);
bitPlane_3 = mod(bitshift(im, -2), 2);
bitPlane_4 = mod(bitshift(im, -3), 2);
bitPlane_5 = mod(bitshift(im, -4), 2);
bitPlane_6 = mod(bitshift(im, -5), 2);
bitPlane_7 = mod(bitshift(im, -6), 2);
bitPlane_8 = mod(bitshift(im, -7), 2);
```

The bitshift(A, -k) function shifts the bits to the right and inserts |k| 0-bits on the left. Bitplanes for all 8 bitplanes are calculated as the modulo of the shifted bits by 2.

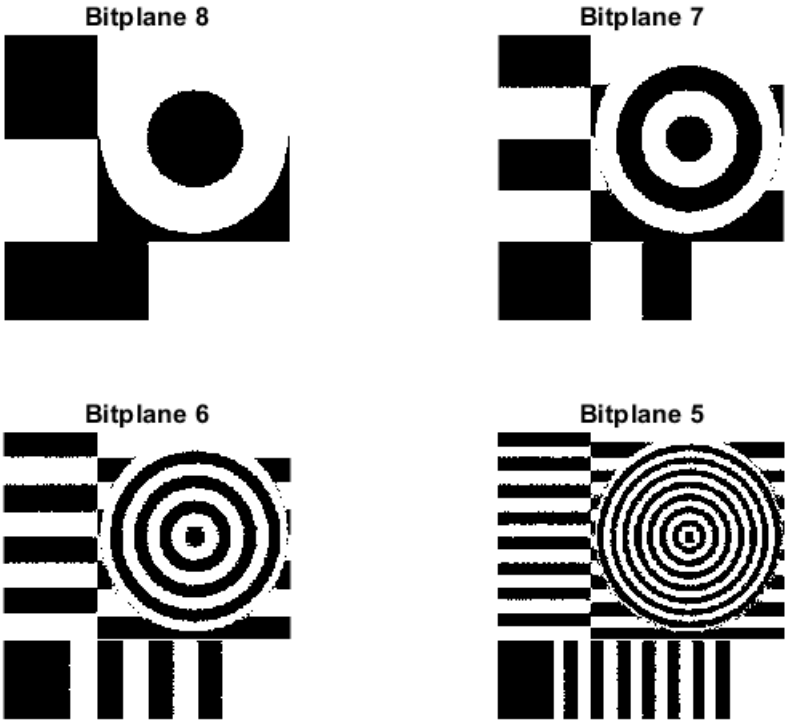
```
% Reconstructing 8 versions of the original image
reconstruction_1 = bitPlane_8 * 128;
reconstruction_2 = bitPlane_8 * 128 + bitPlane_7 * 64;
reconstruction_3 = bitPlane_8 * 128 + bitPlane_7 * 64 + bitPlane_6 * 32 ;
reconstruction_4 = bitPlane_8 * 128 + bitPlane_7 * 64 + bitPlane_6 * 32 + ...
    bitPlane_5 * 16 ;
reconstruction_5 = bitPlane_8 * 128 + bitPlane_7 * 64 + bitPlane_6 * 32 + ...
    bitPlane_5 * 16 + bitPlane_4 * 8 ;
reconstruction_6 = bitPlane_8 * 128 + bitPlane_7 * 64 + bitPlane_6 * 32 + ...
    bitPlane_5 * 16 + bitPlane_4 * 8 + bitPlane_3 * 4 ;
reconstruction_7 = bitPlane_8 * 128 + bitPlane_7 * 64 + bitPlane_6 * 32 + ...
    bitPlane_5 * 16 + bitPlane_4 * 8 + bitPlane_3 * 4 + bitPlane_2 * 2;
reconstruction_8 = bitPlane_8 * 128 + bitPlane_7 * 64 + bitPlane_6 * 32 + ...
    bitPlane_5 * 16 + bitPlane_4 * 8 + bitPlane_3 * 4 + bitPlane_2 * 2 + ...
    bitPlane_1 * 1;
```

Most significant bit: bitPlane_8 contains the highest order bit and bitPlane_1 contains the lowest order bit of all the pixels in the image. Hence, bitPlane_8 is the most significant (upper) bit plane, bitPlane_7 is the second most significant bit plane and so on. In order to reconstruct the image using bit-planes, the nth plane in the pixel is multiplied by the constant 2^n-1. ie. for upper bit plane n = 8, constnt = 2^(8-1) = 128.

```
% Figure 1 - Upper 4 bit-planes
if true
    figure(1),
    subplot(2,2,1); imshow(bitPlane_8, [ ]);title('Bitplane 8');
    subplot(2,2,2); imshow(bitPlane_7, [ ]);title('Bitplane 7');
    subplot(2,2,3); imshow(bitPlane_6, [ ]);title('Bitplane 6');
```

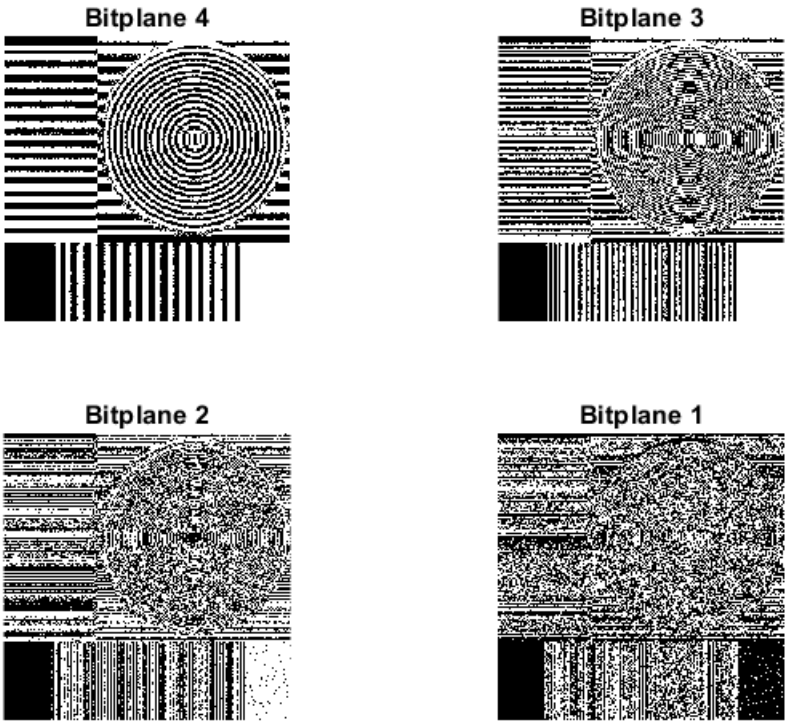
```
    subplot(2,2,4); imshow(bitPlane_5, [ ]);title('Bitplane 5');

end
```



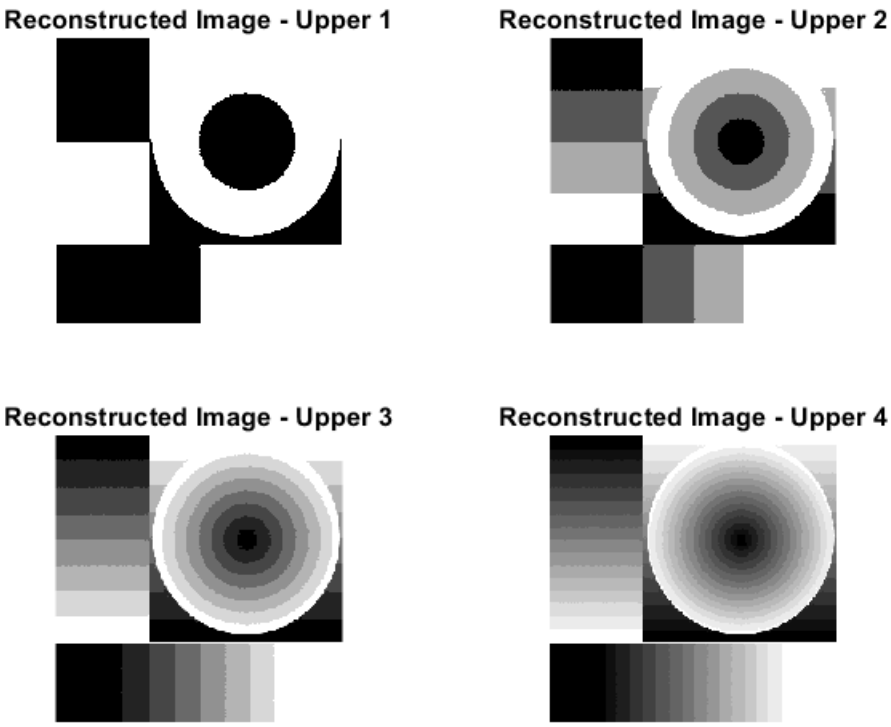

```
% Figure 2 - Lower 4 bit-planes
if true
    figure(2),

    subplot(2,2,1); imshow(bitPlane_4, [ ]);title('Bitplane 4');
    subplot(2,2,2); imshow(bitPlane_3, [ ]);title('Bitplane 3');
    subplot(2,2,3); imshow(bitPlane_2, [ ]);title('Bitplane 2');
    subplot(2,2,4); imshow(bitPlane_1, [ ]);title('Bitplane 1');
end
```



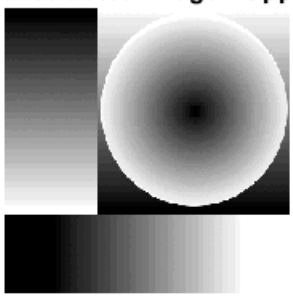

```
% Figure 3 - Reconstructed images - Upper 4 bit-planes
if true
    figure(3),
    subplot(2,2,1); imshow(reconstruction_1, [ ]);title('Reconstructed Image - Upper 1');
    subplot(2,2,2); imshow(reconstruction_2, [ ]);title('Reconstructed Image - Upper 2');
    subplot(2,2,3); imshow(reconstruction_3, [ ]);title('Reconstructed Image - Upper 3');
    subplot(2,2,4); imshow(reconstruction_4, [ ]);title('Reconstructed Image - Upper 4');

end
```

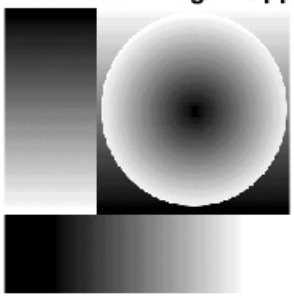



```
% Figure 4 - Reconstructed images - Upper 4 bit-planes
if true
    figure(4),
    subplot(2,2,1); imshow(reconstruction_5, [ ]);title('Reconstructed Image - Upper 5');
    subplot(2,2,2); imshow(reconstruction_6, [ ]);title('Reconstructed Image - Upper 6');
    subplot(2,2,3); imshow(reconstruction_7, [ ]);title('Reconstructed Image - Upper 7');
    subplot(2,2,4); imshow(reconstruction_8, [ ]);title('Reconstructed Image - Upper 8');
end
```

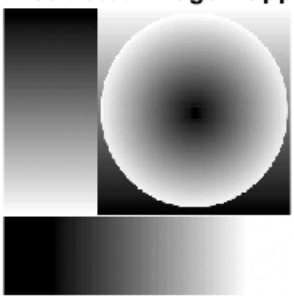
Reconstructed Image - Upper 5



Reconstructed Image - Upper 6



Reconstructed Image - Upper 7



Reconstructed Image - Upper 8