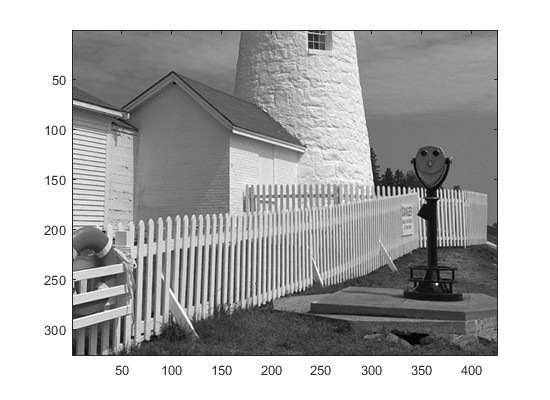
**USMAN MARUF**

**3041120**

**COMPUTER ENGINEERING**

3.1.a

The downsamled image has jagged images and false details relative to the original image.



Original Image



Downsampled image by a factor of 2

3.2.a

% Create the original vector xr1

xr1 = (-2).^(0:6);

% Get the length of xr1

L = length(xr1);

% Create the indexing vector nn

nn = ceil((0.999:1:4\*L)/4);

% Create the zero-order hold version xr1hold

xr1hold = xr1(nn);

% Plot both xr1 and xr1hold for comparison

figure;

% Plot xr1

subplot(2,1,1);

stem(0:6, xr1, 'b', 'filled');

title('Original Signal xr1');

xlabel('Index');

ylabel('Value');

grid on;

% Plot xr1hold

subplot(2,1,2);

stem(0:length(xr1hold)-1, xr1hold, 'r', 'filled');

title('Zero-Order Hold Signal xr1hold');

xlabel('Index');

ylabel('Value');

grid on;

% Display the vectors for verification

disp('xr1:');

disp(xr1);

disp('nn:');

disp(nn);

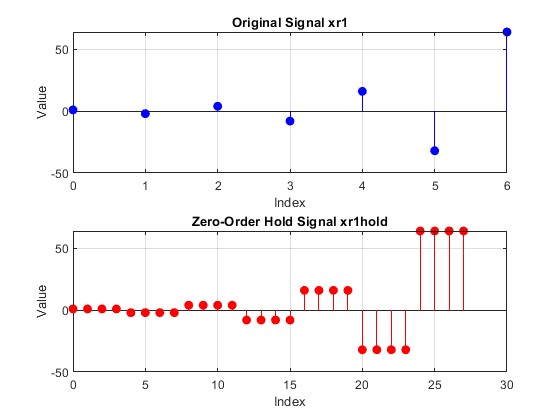
disp('xr1hold:');

disp(xr1hold);

% Calculate and display the interpolation factor

interp\_factor = length(xr1hold) / length(xr1);

fprintf('Interpolation factor: %d\n', interp\_factor);



The indexing vector nn is designed to repeat each index of xr1 four times.

If we consider xr1hold as an interpolated version of xr1, the interpolation factor is 4. This is because each value from xr1 is repeated 4 times in xr1hold. The length of xr1hold (28) is 4 times the length of xr1 (7).

3.2.b

L = length(xx3);

nn = ceil((0.999:1:3\*L)/3);

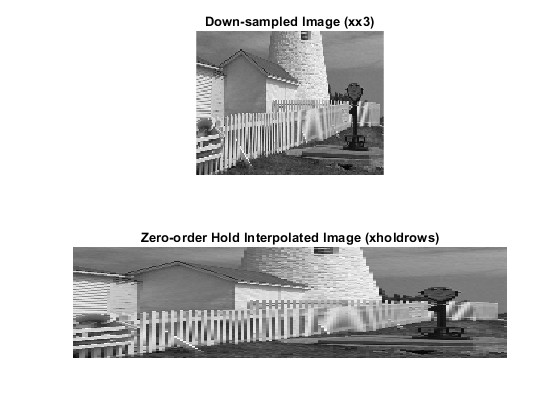
xrholdrows = xx3(:,nn);

figure;

imshow(xx3);

title('Downsampled image');

disp('Size of xx3:'); disp(size(xx3)); disp('Size of xholdrows:'); disp(size(xholdrows));



Size of xx3:

109 142

Size of xholdrows:

109 426

The interpolated image is stretched because its column dimension has been increased by the interpolation factor (3).

3.2.c

L = size(xx3,1);

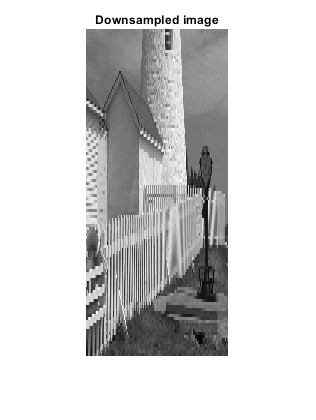
nn = ceil((0.999:1:3\*L)/3);

xrhold = xx3(nn,:);

figure;

imshow(xrhold);

title('Downsampled image');



3.2.d

To determine the interpolation factor when converting xr1 to xr1linear, we need to look at the step sizes of the original and interpolated sequences.

Original sequence (n1): n1 = 0:6

Interpolated sequence (tti): tti = 0:0.1:6

The interpolation factor is the ratio of the step sizes:

Interpolation factor = Original step size / Interpolated step size = 1 / 0.1

= 10

3.2.e

load lighthouse;

xx3 = xx(1:3:end,1:3:end);

[original\_rows, original\_cols] = size(xx);

[rows, cols] = size(xx3);

x = 1:cols;

y = 1:rows;

xi = linspace(1, cols, original\_cols);

yi = linspace(1, rows, original\_rows);

temp = interp1(x, xx3', xi)';

xxlinear = interp1(y, temp, yi);

imshow(xxlinear)

title("interp1 image")



3.2.f

The interpolated image and the original image will have the same resolution. However, xxlinear is not as sharp compared to the original image.

The reconstruction can reduce aliasing effects but cannot remove them completely.

3.2.g

Edges and fence posts are high-frequency features but backgrounds are low frequency features. Linear interpolation produce overall smoother images which is in contrast to blocky images produced by zero-order hold interpolation. This difference is visible in diagonal lines and curves. Also, interpolation makes edges appear smoother but slightly blurred but zero-order hold preserve the sharpness of edges but may make them look jagged or stepped.