# IP - EXPERIMENT No. 2

- Name: Sachi Shah
- Roll No.: C094
- Batch: EB1
- Sap Id: 70321018081

Aim:

1. Apply spatial resolution on the test image to obtain the new image. Use various scaling factors to zoom in and zoom out.
2. Apply amplitude resolution on the test image to obtain new image.

## Theory:

A resolution can be defined as the total number of pixels in an image. Size of an image is given by N X M x m where m is the no. of bits used to represent each pixel, N represents number of rows and M represents number of columns.

Spatial resolution: Number of rows and columns in an image specifies the spatial resolution of an image. If N=8 and M=8 then there will be 8 X8 =64 pixels in that image.

Grey level resolution: The range of grey level in any image depends on number of bits per pixel. Example, if m=8 then there will be 2^8 = 256 shades of grey ranging from 0 to 255.

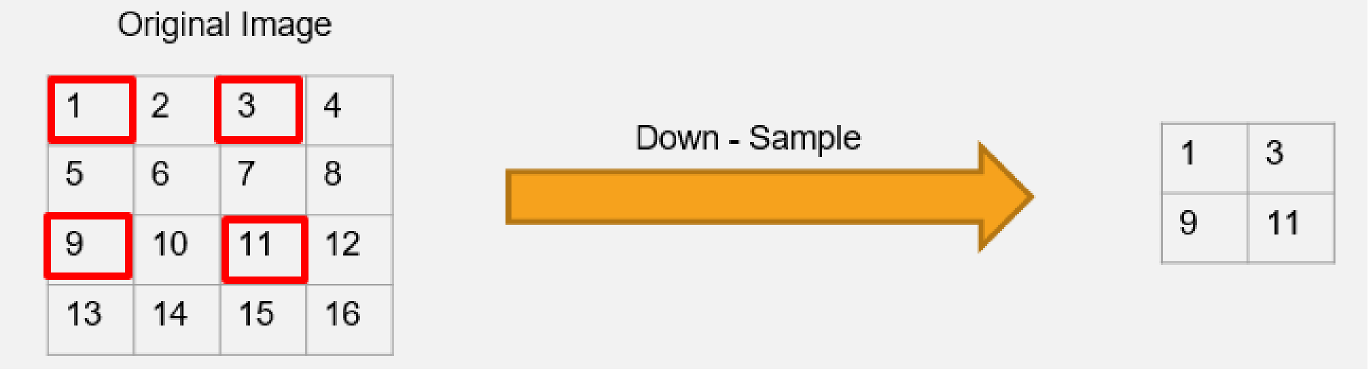
## Spatial Resolution:

# Downsampling :

In the down-sampling technique, the number of pixels in the given image is reduced depending on the sampling frequency. Due to this, the resolution and size of the image decrease.

If sampling frequency =2, output every 2nd element row wise and column wise

Example:



|  |  |  |  |
|---|---|---|---|
| 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 |
| 9 | 10 | 11 | 12 |
| 13 | 14 | 15 | 16 |

|  |  |
|---|---|
| 1 | 3 |
| 9 | 11 |

```
# Import libraries
from skimage import io
import numpy as np
import matplotlib.pyplot as plt
```

```
# read the cameraman.tif image as img1
img1 = io.imread('cameraman.tif')    #reading and saving the image as array in 'img1'
img1_DScopy = img1.copy()

# find the number of rows and columns of this image
r,c = img1.shape    #storing rows and columns in r and c

#Provide Sampling Frequency
f = int(input('Enter sampling Frequency: '))
```

```
Enter sampling Frequency: 2
```

```
# create a array of zeros of half the image size with variable namse img2
DI = np.zeros((r//f,c//f),dtype=np.int)   #DI is the new image array

```

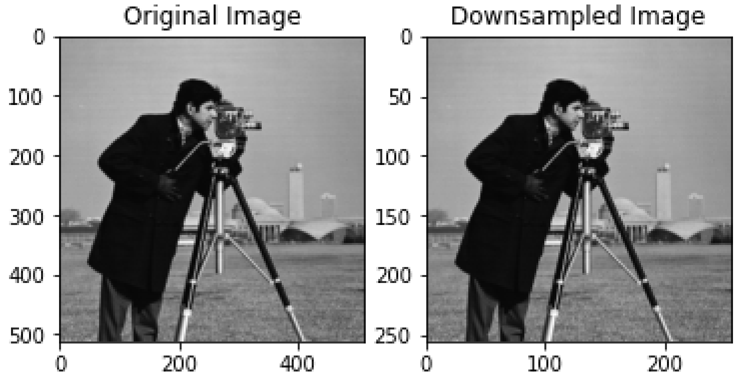
```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: DeprecationWarning: `
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/re
```

```
#write a for loop and fill the alternate pixels of img1 in img2
for i in range(0,r,f):
    for j in range(0,c,f):
        DI[i//f,j//f] = img1[i,j]

```

```
#Plot the original and the down sampled image

plt.subplot(121)
plt.imshow(img1_DScopy,cmap = 'gray')
plt.title('Original Image')

```

```python
plt.subplot(122)
plt.imshow(DI,cmap = 'gray')
plt.title('Downsampled Image')
```

```
Text(0.5, 1.0, 'Downsampled Image')
```



```python
# print he size of original and the downsampled image
print("Size of original image= ",img1_DScopy.shape)
print("Size of downsampled image= ",DI.shape)
```

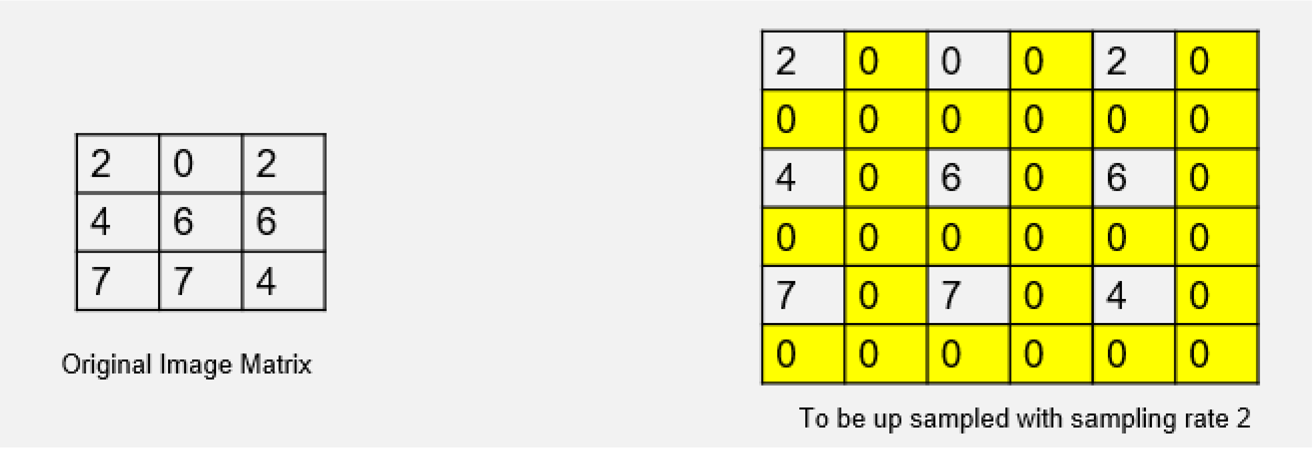
```
Size of original image=  (512, 512)
Size of downsampled image=  (256, 256)
```

## Upsampling

The number of pixels in the down-sampled image can be increased by using up-sampling interpolation techniques. The up-sampling technique increases the resolution as well as the size of the image.

Perform upsampling by zero stuffing

## Example



| | | |
|---|---|---|
| 2 | 0 | 2 |
| 4 | 6 | 6 |
| 7 | 7 | 4 |

Original Image Matrix

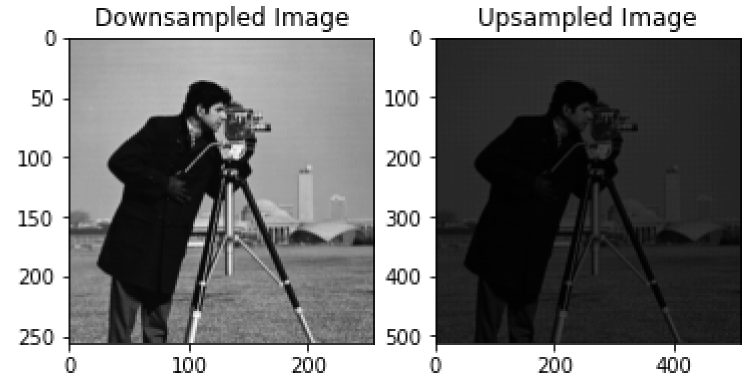
| | | | | | |
|---|---|---|---|---|---|
| 2 | 0 | 0 | 0 | 2 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 6 | 0 | 6 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 7 | 0 | 4 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 |

To be up sampled with sampling rate 2

```python
# Up sampling
```

```
#make an array of zeros of size of original image. name this array as img3.
r1,c1 = DI.shape
f = 2

UI = np.zeros((r1*f,c1*f),dtype = np.int)

# fill the alternate value of the img3 array with the value of img2

for i in range (0,r1*f,f):
    for j in range(0,c1*f,f):
        UI[i,j] = DI[i//f,j//f]


# plot the upsampled image
plt.subplot(121)
plt.imshow(DI,cmap = 'gray')
plt.title('Downsampled Image')

plt.subplot(122)
plt.imshow(UI,cmap = 'gray')
plt.title('Upsampled Image')

```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: DeprecationWarning: `
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/r

Text(0.5, 1.0, 'Upsampled Image')
```



## Amplitude (Intensity) Resolution

It refers to the number of intensity levels used to represent the image. The more intensity levels used, the finer the level of detail discernible in an image. Intensity level resolution is usually given in terms of the number of bits used to store each intensity level.

To reduce the intensity levels from 8 bit to b bit This can be done as follows: if x is the original pixel (of 8 bit image) and
y is the new pixel value (of b bit image)

If 255 --> 2^b

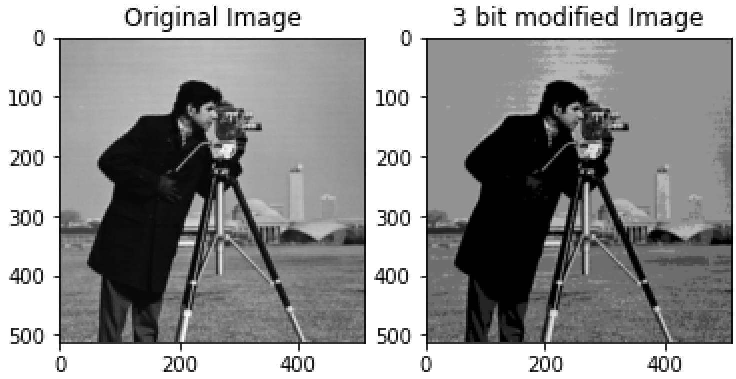
x --> y

Thus new pixel value

y = (x * 2^b) / 255

```
# read the cameraman.tiff file as img2
img2 = io.imread('cameraman.tif')    #reading and saving the image as array in 'img2'
img2_copy = img2.copy()

b = int(input("No. of bits = "))

r,c = img2.shape

# make a array of zeros of size same as the original image img2

# write a for loop and replace each pixel value of img2 with img2*2^b/255
for i in range(r):
    for j in range(c):
        img2[i,j] = img2[i,j] * ((2**b-1)/255)

# plot the original image and the modified image
plt.subplot(121)
plt.imshow(img2_copy,cmap = 'gray')
plt.title('Original Image')

plt.subplot(122)
plt.imshow(img2,cmap = 'gray')
plt.title(f'{b} bit modified Image')
```

```
No. of bits = 3
Text(0.5, 1.0, '3 bit modified Image')
```



```
# check the maximum value of img 1 and img2
print("Maximum value in original = ",img2_copy.max())
print(f"Maximum value in {b} bit modified image = ",img2.max())

```

```
Maximum value in original =  255
Maximum value in 3 bit modified image =  7
```

# Conclusion

1. We implemented Spatial Resolution.
2. We scaled up the physical size of the image using upsampling using zero stuffing method with different sampling factors.
3. However, the image quality of zero stuffing upsampling is poor and can be improved using averaging.
4. We reduced the size (i.e spatial resolution) by down sampling.
5. Example: Downsampling by factor of 2 - select every alternate value of original image.
6. We implemented amplitude resolution and observed the reduction in grey levels from [0,255] to [0,7].