

IIVP Lab Practice Experiments

Instructor: Prof. Anupam

Date: 04.09.2020

TAs: GC Jana

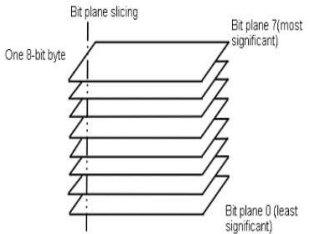
Instructions: Use MATLAB and/or Python and/or Octave tools for the following questions. Do not use the inbuilt functions unless mentioned in the question. If input image not give or specified then you can use Lenna image (popular picture use for image processing) as a sample image.


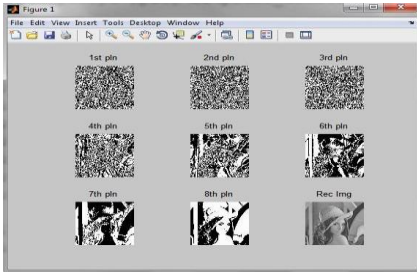
Topics Covered in the Lecture Session: Arithmetic operations, image negative, Thresholding of an Image, contrast stretching, bit plan slicing, zooming by interpolation and replication and resizing by the nearest neighbor concept, Spatial and Frequency domain analysis.





Covered in Previous Lab Session: [Arithmetic operations](#), [image negative](#), [Thresholding of an Image](#).

Aim of this Lab Session: Bit Plane Slicing, Histogram Specification,

~ * * * ~

Experiment No.1	Related to Bit Plane Slicing
Aim	Understand the importance of bit plane slicing in image enhancement & image compression.
Tools and Library	Python, without using library involved in bitwise operation.
Theory/Hint	<p>This transformation involves determining the number of usually significant bits in an image. In case of a 8 bit image each pixel is represented by 8 bits. Imagine that the image is composed of eight 1 bit planes ranging from bit plane 0 for the least significant bit to bit plane 7 for the most significant bit. Plane 0 contains all the lowest order bits in the bytes comprising the pixels in the image & plane 7 contains all the high order bits. The higher order bits contain usually significant data and the other bit planes contribute to more subtle details in the image. Separating a digital image into its bit planes is useful for analyzing the relative importance played by each bit of the image.</p> 
Question	Load an input image (grayscale) and extract each bit using bitwise AND operation. Display all the bit planes formed by bits extracted. Also, reconstruct the input image using extracted bit planes.
Algorithmic Steps:	<ol style="list-style-type: none"> 1. Read i/p image 2. Use bitand operation to extract each bit 3. Do the step 2 for every pixel. 4. Display the original image and the biplanes formed by bits extracted

Input and possible Output Images	<div></div> <div></div> <div>Input image of Exp. 1</div> <div>Possible Output after performing given experiment</div>																																				
Experiment Shows	Higher order bit planes carry maximum visual information																																				
Experiment No. 2	Related to Histogram Specification																																				
Aim	Understand the histogram specification techniques in continuous domain.																																				
Tools and Library	Python (without using callable library)																																				
Theory/Hint	<p>Histogram equalization automatically determines a transformation function that seeks to produce an output image that has a uniform histogram. But it is useful sometimes to be able specify the shape of the histogram that we wish the processed image to have. The method used to generate a processed image that has a specified histogram is called histogram specification.</p> $S_k = T(r_k) = \sum \Pr(r_j) \quad k = 0, 1, 2, 3, \dots, L-1$ $V_k = G(z_k) = \sum P_z(z_j) \quad k = 0, 1, 2, 3, \dots, L-1$ $Z_k = G^{-1}(T(r_k)) \quad k = 0, 1, 2, 3, \dots, L-1$ <p>Map each pixel with level r_k into a corresponding pixel with level S_k. Obtain the transformation function G from a given histogram $P_z(z)$. For any Z_q this transformation function yields a corresponding value V_q. We would find the corresponding value Z_q from G^{-1}.</p>																																				
Algorithmic Steps:	<ol style="list-style-type: none">1. Obtain the histogram of the given image.2. Map each level r_k to S_k3. Obtain the transformation function G from the given $P_z(z)$4. Calculate Z_k for each value of S_k5. For each pixel in the original image, if the value of that pixel is r_k, map this value to its corresponding level S_k, then map level S_k into the final value Z_k6. Display the modified image and its histogram																																				
Question	<p>Given original image (a) and targeted image (b) performer Histogram Specification using followings and show the output image.</p> <p style="text-align: center;">Original Image Grey Level Distribution</p> <table><tr><td>Grey Level (r)</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr><tr><td>Number of Pixels(p)</td><td>8</td><td>10</td><td>10</td><td>2</td><td>12</td><td>16</td><td>4</td><td>2</td></tr></table> <p style="text-align: center;">Target Image Grey Level Distribution</p> <table><tr><td>Grey Level(r)</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr><tr><td>Number of Pixels(p)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>20</td><td>20</td><td>16</td><td>8</td></tr></table>	Grey Level (r)	0	1	2	3	4	5	6	7	Number of Pixels(p)	8	10	10	2	12	16	4	2	Grey Level(r)	0	1	2	3	4	5	6	7	Number of Pixels(p)	0	0	0	0	20	20	16	8
Grey Level (r)	0	1	2	3	4	5	6	7																													
Number of Pixels(p)	8	10	10	2	12	16	4	2																													
Grey Level(r)	0	1	2	3	4	5	6	7																													
Number of Pixels(p)	0	0	0	0	20	20	16	8																													

Input and possible output image	 Original image (a)	 Target images (b)
Experiment No. 3	Filtering in spatial domain: Low pass filtering	
Aim	To implement low pass filtering in spatial domain	
Tools and Library	Python (without using callable library)	
Question	Load the given input image, Ignore the border pixel, Apply low pass mask to each and every pixel. Display the o/p image.	
Input and possible output images	 Input image of Exp.3	 Output image after performing Low pass filter
Experiment Shows	Low pass filtering makes the image blurred.	
Experiment No. 4	Filtering in spatial domain: High pass filtering	
Question	Load the given input image, Ignore the border pixel and apply high pass mask to each and every pixel. Display the o/p image. Also, Show that high pass= Original – low pass. Note: Use the given input image of Experiment No.3.	
Experiment Shows	High pass filtering makes the image sharpened.	
Experiment No. 5	Zooming by interpolation and replication	
Algorithmic Steps:	Replication: <ol style="list-style-type: none"> 1. Read i/p image. 2. Replicate each pixel 3. Replicate each row 4. Display o/p image Interpolation <ol style="list-style-type: none"> 1. Read i/p image 2. Average of two adjacent pixels along the rows is taken and placed between two pixels. 3. Do the same along columns 4. Display o/p image. 	
Experiment Shows	Zooming by interpolation is more effective than zooming by replication	