|  |  |  |  |
| --- | --- | --- | --- |
| **Course Name:** | **Digital Image Processing** | **Semester:** | **VII** |
| **Date of Performance:** |  | **Batch No:** |  |
| **Faculty Name:** |  | **Roll No:** |  |
| **Faculty Sign & Date:** |  | **Grade/Marks:** |  |

**Experiment No: 3**

**Title: To study process Contrast stretching**

|  |
| --- |
| **Aim and Objective of the Experiment:** |
| To study the process contrast streching |

|  |
| --- |
| **COs to be achieved:** |
| 1. **Understand fundamental theory and models of image processing** |

|  |
| --- |
| **Theory:** |
| Normalization is commonly used to improve the contrast in an image without distorting relative gray-level intensities too significantly.  We begin by considering an image  [wom1](https://homepages.inf.ed.ac.uk/rbf/HIPR2/images/wom1.gif)  which can easily be enhanced by the most simple of contrast stretching implementations because the intensity histogram forms a tight, narrow cluster between the Gray level intensity values of 79 - 136, as shown in  [wom1hst1](https://homepages.inf.ed.ac.uk/rbf/HIPR2/images/wom1hst1.gif)  After contrast stretching, using a simple linear interpolation between *c = 79* and *d = 136*, we obtain  [wom1str1](https://homepages.inf.ed.ac.uk/rbf/HIPR2/images/wom1str1.gif)  Compare the histogram of the original image with that of the contrast-stretched version  [wom1hst2](https://homepages.inf.ed.ac.uk/rbf/HIPR2/images/wom1hst2.gif)  While this result is a significant improvement over the original, the enhanced image itself still appears somewhat flat. Histogram equalizing the image increases contrast dramatically, but yields an artificial-looking result  [wom1heq1](https://homepages.inf.ed.ac.uk/rbf/HIPR2/images/wom1heq1.gif)  In this case, we can achieve better results by contrast stretching the image over a more narrow range of graylevel values from the original image. For example, by setting the cutoff fraction parameter to 0.03, we obtain the contrast-stretched image  [wom1str2](https://homepages.inf.ed.ac.uk/rbf/HIPR2/images/wom1str2.gif)  and its corresponding histogram  [wom1hst3](https://homepages.inf.ed.ac.uk/rbf/HIPR2/images/wom1hst3.gif)  Note that this operation has effectively spread out the information contained in the original histogram peak (thus improving contrast in the interesting face regions) by pushing those intensity levels to the left of the peak down the histogram *x*-axis towards 0. Setting the cutoff fraction to a higher value, *e.g.* 0.125, yields the contrast stretched image  [wom1str3](https://homepages.inf.ed.ac.uk/rbf/HIPR2/images/wom1str3.gif)  As shown in the histogram  [wom1hst4](https://homepages.inf.ed.ac.uk/rbf/HIPR2/images/wom1hst4.gif)  most of the information to the left of the peak in the original image is mapped to 0 so that the peak can spread out even further and begin pushing values to its right up to 255.  As an example of an image which is more difficult to enhance, consider  [moo2](https://homepages.inf.ed.ac.uk/rbf/HIPR2/images/moo2.gif)  which shows a low contrast image of a lunar surface.  The image  [moo2hst2](https://homepages.inf.ed.ac.uk/rbf/HIPR2/images/moo2hst2.gif)  shows the intensity histogram of this image. Note that only part of the *y*-axis has been shown for clarity. The minimum and maximum values in this 8-bit image are 0 and 255 respectively, and so straightforward normalization to the range 0 - 255 produces absolutely no effect. However, we *can* enhance the picture by ignoring all pixel values outside the 1% and 99% percentiles, and only applying contrast stretching to those pixels in between. The outliers are simply forced to either 0 or 255 depending upon which side of the range they lie on.  [moo2str1](https://homepages.inf.ed.ac.uk/rbf/HIPR2/images/moo2str1.gif)  shows the result of this enhancement. Notice that the contrast has been significantly improved. Compare this with the corresponding enhancement achieved using histogram equalization.  Normalization can also be used when converting from one [image type](https://homepages.inf.ed.ac.uk/rbf/HIPR2/value.htm) to another, for instance from floating point pixel values to 8-bit integer pixel values. As an example the pixel values in the floating point image might run from 0 to 5000. Normalizing this range to 0-255 allows easy conversion to 8-bit integers. Obviously, some information might be lost in the compression process, but the relative intensities of the pixels will be preserved. |

|  |
| --- |
| **Stepwise-Procedure:** |
| 1. Click selfie through your phone 2. Get image into the PC on which you are working using gmail or any other medium 3. Read image using MATLAB online   A=imread(“Path of the image”)   1. Covert image to grayscale 2. Plot histogram using imhist command 3. Find min\_val and max\_val from histogram 4. Covert image datatype to double 5. Bring all values between zero to one (Normalize) 6. Multiply all elements to range between 0 to 255 7. Round-off and Convert data back to unit 8 8. Use imashowpair or montage to display both images side by side 9. Repeat same operation with color images 10. Split 3 color planes 11. Repeat contrast stretching of grayscale on each color plane and merge final output to for color channel 12. Use imadjust on individual channel and compare it with contract stretching outputs after reconstructing RGB image 13. Repeat same on python |

|  |
| --- |
| **Output** |
| Upload picture screenshots for all approaches with intermediate steps |

|  |
| --- |
| **Conclusions:** |
|  |

|  |
| --- |
| **Post Lab Subjective/Objective type Questions:** |
| Answer the following questions:   1. What is background operation of imadjust. 2. Is any of the methods used in above experiment leads to color change? if yes, why? |

|  |
| --- |
| **Signature of faculty in-charge with Date:** |