Shri Ramdeobaba College of Engineering and Management, Nagpur

Department of Electronics Engineering

Digital Image Processing (ENT 355-3)

Name: Prajwal Pandurang Shette

Roll No: B1- 12

Evaluation 2

Aim: Write a python code to apply local image enhancement on an image using Histogram processing

Theory: Local Enhancement Histogram processing methods are global processing, in the sense that pixels are modified by a transformation function based on the gray-level content of an entire image. Histogram processing methods are global processing, in the sense that pixels are modified by a transformation function based on the gray-level content of an entire image.

Sometimes, we may need to enhance details over small areas in an image, which is called a local enhancement.

Sometimes, we may need to enhance details over small areas in an image, which is called a local enhancement.

Local Enhancement define a square or rectangular neighborhood and move the center of this area from pixel to pixel. define a square or rectangular neighborhood and move the center of this area from pixel to pixel. at each location, the histogram of the points in the neighborhood is computed and either histogram equalization or histogram specification transformation function is obtained. at each location, the histogram of the points in the neighborhood is computed and either histogram equalization or histogram specification transformation function is obtained. another approach used to reduce computation is to utilize nonoverlapping regions, but it usually produces an undesirable checkerboard effect. another approach used to reduce computation is to utilize nonoverlapping regions, but it usually produces an undesirable checkerboard effect. a)Original image (slightly blurred to reduce noise) b)global histogram equalization (enhance noise & slightly

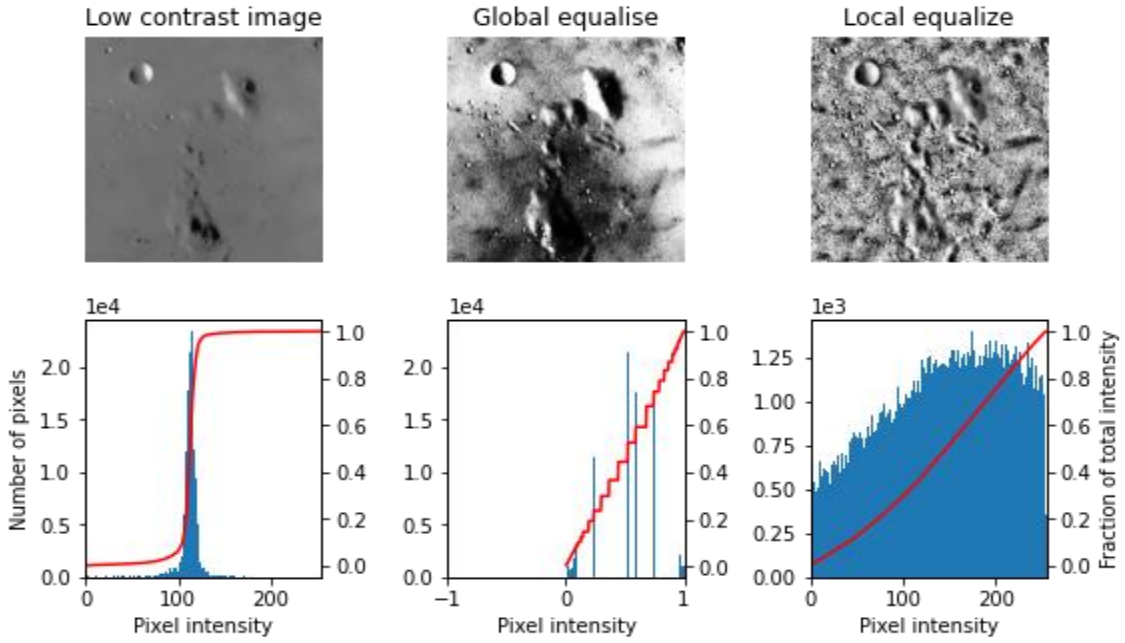
increase contrast but the construction is not changed) c)local histogram equalization using 7x7 neighborhood (reveals the small squares inside larger ones of the original image.

Code:

```
import numpy as np
import matplotlib
import matplotlib.pyplot as plt

from skimage import data
from skimage.util.dtype import dtype_range
from skimage.util import img_as_ubyte
from skimage import exposure
from skimage.morphology import disk
from skimage.morphology import ball
from skimage.filters import rank


matplotlib.rcParams['font.size'] = 9


def plot_img_and_hist(image, axes, bins=256):
    """Plot an image along with its histogram and cumulative histogram.

    """
    ax_img, ax_hist = axes
    ax_cdf = ax_hist.twinx()

    # Display image
    ax_img.imshow(image, cmap=plt.cm.gray)
    ax_img.set_axis_off()

    # Display histogram
    ax_hist.hist(image.ravel(), bins=bins)
    ax_hist.ticklabel_format(axis='y', style='scientific', scilimits=(0, 0))
    ax_hist.set_xlabel('Pixel intensity')

    xmin, xmax = dtype_range[image.dtype.type]
    ax_hist.set_xlim(xmin, xmax)

    # Display cumulative distribution
    img_cdf, bins = exposure.cumulative_distribution(image, bins)
    ax_cdf.plot(bins, img_cdf, 'r')

    return ax_img, ax_hist, ax_cdf
```

```
# Load an example image
img = img_as_ubyte(data.moon())

# Global equalize
img_rescale = exposure.equalize_hist(img)

# Equalization
footprint = disk(30)
img_eq = rank.equalize(img, footprint=footprint)


# Display results
fig = plt.figure(figsize=(8, 5))
axes = np.zeros((2, 3), dtype=object)
axes[0, 0] = plt.subplot(2, 3, 1)
axes[0, 1] = plt.subplot(2, 3, 2, sharex=axes[0, 0], sharey=axes[0, 0])
axes[0, 2] = plt.subplot(2, 3, 3, sharex=axes[0, 0], sharey=axes[0, 0])
axes[1, 0] = plt.subplot(2, 3, 4)
axes[1, 1] = plt.subplot(2, 3, 5)
axes[1, 2] = plt.subplot(2, 3, 6)

ax_img, ax_hist, ax_cdf = plot_img_and_hist(img, axes[:, 0])
ax_img.set_title('Low contrast image')
ax_hist.set_ylabel('Number of pixels')

ax_img, ax_hist, ax_cdf = plot_img_and_hist(img_rescale, axes[:, 1])
ax_img.set_title('Global equalise')

ax_img, ax_hist, ax_cdf = plot_img_and_hist(img_eq, axes[:, 2])
ax_img.set_title('Local equalize')
ax_cdf.set_ylabel('Fraction of total intensity')


# prevent overlap of y-axis labels
fig.tight_layout()
```

Output:



Conclusion: We learned about how to apply local image enhancement on an image using Histogram processing.