Assignment2: Histogram Modification and Color Processing

Pariyakorn Chuisakun

Department of Computer Science and Engineer

University of South Florida

1. INTRODUCTION

This assignment focuses on image processing, emphasizing histogram modification, color processing, and utilizing the OpenCV library. It primarily focuses on managing image Regions of Interest (ROIs) and executing various histogram modifications and color processing on these ROIs to achieve distinct effects and improvements in image quality and details.

2. IMPLEMENTATION

The immediate approach to the implementation was to modify the previous utility functions and code to accept more parameters. Most of the image processing in the implementation was done utilizing OpenCV functions. The code is given a parameter text file where it reads through to retrieve information about the input image, output image, whether to use OpenCV, functions to be applied, and their function specific parameters.

Histogram Stretching

The utility function, histogramStretch, performs a histogram stretching operation on an input image. This operation expands the intensity range of the input image, which can enhance the contrast and bring out features in images that are initially either too dark or too light.

The function begins by checking if the source image has three channels, if it does it converts it to gray scale. If it is already in grayscale, it clones the source image to work on it without altering the original data. The function determines the minimum (**Imin**) and maximum (**Imax**) intensity values present in the grayscale image using OpenCV's **minMaxLoc** function. Next, values **c** and **d** are determined, where **c** is slightly above the minimal intensity and **d** is slightly below the maximum intensity in the original image. Then, contrast stretching is applied. The pixel intensity values below **c** are set to **A**, those above **d** to **B**, and those in between are scaled linearly between **A** and **B**. Histograms of both the original and stretched images are calculated using the **calcHist** function and plotted into **histImageOrig** and **histImageStretch** respectively. The stretched image is saved into the target output and histograms of the original and stretched images are saved.

Histogram Equalization

The **histogramEqualization** function is applied to an input image, adjusting its pixel intensity distribution such that the resulting image's histogram is approximately flat.

The function first utilizes the cv_gray function to convert the image to grayscale if it is not already. The function applies OpenCV's **equalizeHist()** function directly to **tgt**, which should contain the grayscale version of **src**. **equalizeHist()** equalizes the histogram of the grayscale image, effectively redistributing pixel intensities to enhance contrast.

Dark Pixel Equalization

The **equalizeDarkPixels** function performs selective histogram equalization on dark pixels of an image, aiming to enhance the contrast in darker regions while preserving the original intensity of brighter regions.

The image is first converted to gray scale. A binary mask is created based on the darkness threshold, where pixels below the threshold are considered true (or 1). The grayscale image is equalized independently,

resulting in a new image. Utilizing the mask, equalized pixel values are assigned only to dark pixels in the original image, then it is saved as the output image.

Augmenting Grey Images

The **greyAugumentedImages** function encompasses three steps in image processing: extraction of a Region of Interest (ROI), application of two distinct histogram enhancement techniques, and rotation of the obtained images.

The ROI is first extracted and saved as the original ROI image. It then rotates the original ROI through three 90-degree increments, saving each. It applies a histogram stretching operation to the original ROI and rotates the resultant image three times, with each rotated version being saved. Then the original ROI undergoes histogram equalization and is subsequently rotated through three angles, with each state being saved.

RGB Equalization

The **equalizeRGB** function aims to perform histogram equalization on specified channels (R, G, B) of a source color image.

It first tries to validate the input to ensure a color image is input, and then extracts its R, G, and B channels. It checks user input to determine which channels to equalize, conducting the operation and saving intermediary results. It then merges the possibly equalized channels and attempts to convert the resultant matrix from HSV to BGR color space before saving.

HSI Equalization

The **equalizeHSI** aims to perform histogram equalization on the HSI (Hue, Saturation, and Intensity) channels of a color image.

It first ensures that the input image is a color image before converting to HSI color space. The cvtcolor() OpenCV functions do not work in HSI color space, so the color image is converted to HSV. It then splits the HSI image into its channels and equalizes them based on user input. Finally it merges the potentially equalized channels, converts the image back to BGR color space, and saves the resultant image

Augmenting Color Images

The **colorAugmentedImages** function aims to perform applies histogram equalization to different color channels and rotations to a Region of Interest (ROI) from a source image.

First, the ROI is extracted, rotated through three 90-degree intervals, and saved. Histogram equalization is applied to the Intensity channel and then rotates the result, saving the images. Then it equalizes the Saturation channel of the ROI and then rotates and saves these images.

3. RESULTING IMAGES AND DISCUSSION

Histogram Stretching

The first image is the original 11.jpg image

used for the comparison.

Above are the resulting images from:

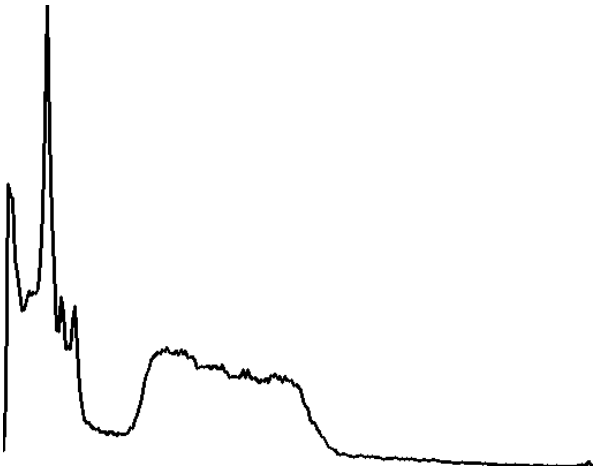
11.jpg 11_gray.jpg opencv gray

11.jpg 11_stretched50_100.jpg opencv stretch 50 100

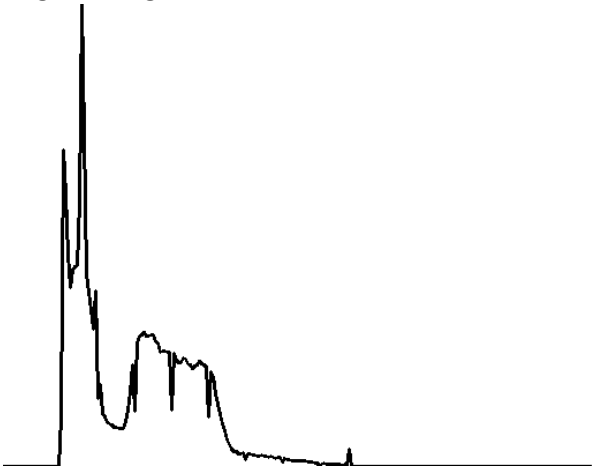
11.jpg 11_stretched0_150.jpg opencv stretch 25 150

Above are the resulting images from:

00



Above is the histogram generated from the original image



Above is the histogram generated after histogram stretching.

Histogram Equalization

Above are the resulting images from:

11.jpg 11_gray.jpg opencv gray

11.jpg 11_equalize.jpg opencv equalize

The **histogramEqualization** function aimed to equalize the entire grayscale image or ROI, enhancing the overall contrast. The output images showed improved visibility and detail.

Dark Pixel Equalization

Above are the resulting images from:

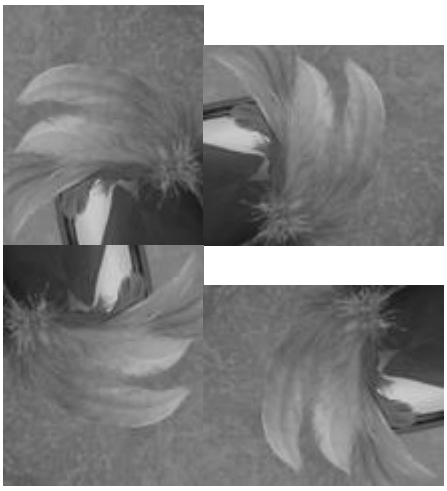
11.jpg 11_equalTh60.jpg opencv equalize_dark 60

11.jpg 11_equalTh150.jpg opencv equalize_dark 150

Function **equalizeDarkPixels** targeted dark pixels (below a defined threshold) for equalization, preserving brighter regions. The utility showed potential for enhancing details in darker areas without overexposing brighter section.

Augmenting Grey Images

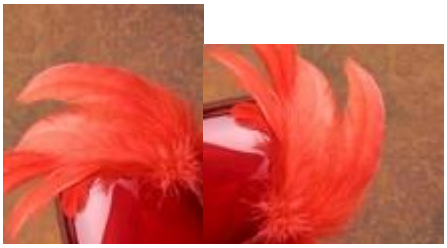
Above are the Rotations of the ROI after histogram equalization.



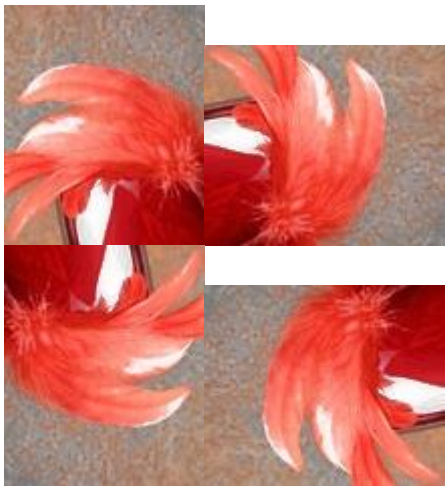
Above are the Rotations of the ROI after histogram stretching.

The **augmentedGreyImages** function provides enhanced and variously rotated images for potential analytical use. The images that were equalized have a much higher contrast than prior. The images that were stretched, have smoother appearance and the details are less sharp.

Augmenting Color Images



Above are the Rotations of the Original ROI.



Above are the Rotations of the ROI after equalizing the S channel.

Above are the Rotations of the ROI after equalizing the I channel.

The **equalizeHSI** function provided options to equalize specific channels in the HSI space. The benefits included improved flexibility without dramatically altering the primary color scheme, although changing the intensity and saturation. Changing the intensity and saturation channel brought out more textures that were not present in the original image.

4. CONCLUSION

In conclusion, the utility functions implemented during this assignment helped me understand how to work with OpenCV functions and better acquaint myself with image augmenting. This assignment built up my toolset for a range of image processing tasks, histogram equalization across different color spaces and channels, dark pixel enhancement, and image augmentation. While testing various images and parameters, I developed a better understanding of how each function affects an image. The output images greatly help visualize the results of each function. Being able to produce a histogram for an image also helps to visualize the effect of histogram stretching on the histogram itself.