TEE 461 Image Processing

Zach Swinnerton

Project 1 App

OVERVIEW:

This MATLAB app is designed to allow the user to load an image to be modified by the GUI sliders. The user can manipulate the brightness/darkness, contrast, and/or sharpness sliders to their desired results. They can then save the modified image to a directory of their choosing for later use.

HOW TO USE:

The user will click the Load Image button in the top left corner of the GUI and select a file from a directory of their choosing. The file must be of type “.tif”, “.png”, or “.jpeg”. Once the image is loaded in the GUI will update with the original image displayed on the left and right axes labeled Original Image and Modified Image. The center axes, called Pixel Mapping, will update to show a pixel mapping for the original image. This mapping will default to the Identity Line from (0,0) to (1,1). The user then can modify any of the sliders to modify the image. When the sliders are moved, the Pixel Mapping graph and the Modified Image axes will update to show the new image and pixel representation for that transformation or combination of transformations. The Sharpening slider will not affect the Pixel Mapping graph and only update the Modified Image axes, since sharpening does not change the pixel mapping. If the user would like to return the Modified Image to the Original Image or reset a certain slider, they must move the slider back to its start position. For the Brightness/Darkness Slider the start position is a gamma value of 1. For the Contrast Slider the start position is 0, and for the Sharpness Slider the start position is 0. Once the user has finished modifying the image, they can save the image to a directory of their choosing by selecting the Save Image button at the top left of the GUI, adjacent to the Load Image button.

IMAGE PROCESSING METHODS:

For the Brightness/Darkness Slider the function I used was:

If gamma (γ) is less than 1 (slider moved to the left) then the function increases brightness. If gamma is greater than 1 (slider moved to the right) then the function decreases brightness. The Pixel Mapping graph shows the updated curve using the same equation but still showing the Identity Line as a dashed blue line. Since increase and decrease contrast are inversely related based on the gamma value the range for increase brightness (1/x) is smaller than the range for decrease brightness (x) or darken. I modified the range of the slider to accommodate for as much of both transformations as reasonably possible.

For the Contrast Slider the functions I used were:

The slider value is input as the curvature and the GUI calculates the endpoint scalar needed to maintain 0 to 0 and 1 to 1 scaling for the Pixel Mapping graph. The horizontal and vertical shifts are 0.5 to shift the function to the proper location. When the slider is moved to the left for decrease contrast the absolute value of the slider value is taken as the input for the curvature scalar. Like the Brightness/Darkness Slider the slider value for the contrast is not evenly distributed. This is because of scaling tan() vs arctan() increase or decrease at different rates. The decrease contrast range is much smaller than the increase contrast range. I modified the range of the slider to accommodate as much increase and decrease as reasonably possible.

For the Sharpness Slider I used the following kernel to sharpen in horizontal, vertical, and horizontal directions:

I used convolution to calculate the edge detected filter and added it to the input image (original or modified) based on a scalar weighting. The scalar used to determine weighting is the sliders value. The Sharpness Slider range is from 0 to 1 since I felt that 1 times magnification of edges was plenty for this application.

Aside from image processing methods simple if/else checks were needed to ensure if the user updated the slider value left or right, after modifying it once, that the new value was correctly reflected. Also, the overall functionality of the GUI is by first checking the current gamma value from the Brightness/Darkness Slider and applying it to the original image, then applying the curvature based on the Contrast Slider value, and finally adding the sharpness weighting based on the Sharpness Slider value. If any of these sliders are not modified the code continues to the next section with the input image passed along to the next transformation. Once complete, the Pixel Mapping and the Modified Image is displayed. If both the Brightness/Darkness Slider and the Contrast Slider are modified, then the Pixel Mapping will display a graph representing a combination of these transforms. The building image and the totem image from class for testing, but any image should work. Practice care when clicking the Load Image button, if the mouse moves quickly to Save Image while clicking Load Image, MATLAB may accidently think the click is intended for Save Image.