TEE 461 Image Processing

Zach Swinnerton

Project 2 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 Hue, Saturation, and Intensity sliders to their desired ranges, and select between 1, 2, and/or 3 colors to segment from the original image. 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 “.jpg”. Once the image is loaded in the GUI will update with the original image displayed on the left and right axes. The user then can modify any of the sliders to modify the image segmentation process. When the sliders are moved the segmented image axes will update to show the new image for that segmentation combination and the slider ranges update to prevent most out of bounds issues. The segmented image will show the current color selected when making modifications to that color’s ranges for H, S, and I. Once the user selects the next color to segment by clicking the next color button at the bottom, the sliders reset to the original values and the segmented axes returns to the original image for the user to easily see how their 2nd and/or 3rd segmentation ranges apply to the image. The segmented axes will also update the title to the corresponding color button selected for the user to clearly see which color ranges they are modifying. The current positions of the sliders are stored for the color segmentation when the user selects to the next color button. This allows the user the ability to go back to previous color segmentations to update if necessary. Once the Segment Button is selected the app will use if else statements to combine the three or fewer segmented images together (depending on how many color values are changed from the default settings). The segmented axes will then update to show the combined segmented image and update the title to match. This image is the final image the user then can save for later use. If the user does not prefer the combined segmented image ranges, they can go back to each color range to update them for better segmentation, then click the Segment Button again to apply these changes and recombine or select the Reset All Button to start from scratch. 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.

NOTE: The user cannot select an upper or lower limit for H, S, or I to be equal to the other limit of that variable. This is because the sliders update the upper and lower limits based on the user’s selection for the slider values. For example, Hue cannot have an upper limit of 1 and a lower limit of 1, since the slider cannot properly update and scale to show a single value on a bar. The user must select a value close to, but not equal to the same value as the other limit for selecting a small range.

IMAGE PROCESSING METHODS:

The main image processing method for this app is the use of the rgb2hsv() function. Then it takes each H, S, and I component and allowing the user to set ranged limits to determine if the pixel equal to that value is preserved, or replaced with a gray pixel [0.5,0.5,0.5]. The app then uses if else statements to check each individual color segmentation to see if it has been modified from the original. If so, then it combines those segmented images together to create the final image.

UPDATES FROM ORIGINAL DESIGN:

I completed this project before submitting the proposal, but later decided to try to improve its speed and versatility. My previous design took about a minute or two to update to the axes which I felt was inadequate. I learned that MATLAB 2021 has improved App Designer’s processing speeds, so older version may result in slower app speeds. I removed redundant if/else checks and for loops to increase the processing speed to take only a few seconds to update and added visual ques to the user when the application is processing. Each Color Button and the Segment Button will update its name with “Processing…” so the user knows that it is working. Once the button name is reset back to its original name the process is complete and displayed. I added the ability for the user to go back and update the previously stored color range values for each color if the resulting image was not what they were expecting. This versatility allows for ease of use and eliminates having to close and re-open the app to try new ranges. I also added the updating segmented image axes title to, again, visualize to the user which color they are changing the ranges for. This eliminates confusion since there can be 1, 2, and/or 3 possible color range variable sets being stored.

I tested this app on the chalk image and the control systems book image, and the color segmentation works as expected.

Graphical user interface

Description automatically generated Graphical user interface, application

Description automatically generated

I was not able to fix the slider scaling occasionally causing overlaps of slider tick labels. This would prove difficult to fix since there is a wide range of possible user input ranges to code around. I left this issue alone and figured users can work around since high precision is not required.