**PAint-By-Number Generator: What’s Happening Here?**

1. **Capture image and convert to CIELAB color space**

The CIELAB color space expresses color as three values: *L\** for perceptual lightness and *a\** and *b\** for the four unique colors of human vision (red, green, blue, and yellow). It is useful in this context because the image should be quantized (step 2) by colors that appear unique to the user.

1. **Color quantization and mapping to unique base colors**

Color quantization refers to reducing the number of unique colors in an image, in this context the captured image is reduced to nine unique colors. This is accomplished using k-means clustering. Next, the base colors that are returned from the quantization are mapped to the most similar colored pencil color that is available.

1. **Masking and finding contours**

Now that the image is grouped into nine unique colors, nine masks are formed. Each mask contains just the elements in the image that are of the same color. For example, the red mask will contain only ‘blobs’ that are red from the original image. Using these masks, the edges of each unique color area is identified and placed into a contour library. This library will be used when making the outlines that need to be colored in.\*

1. **Finding label locations**

Perhaps the most challenging part of this project was figuring out where in each ‘blob’ to place the number that maps to the correct color. Simply placing it in the center does not work because this is not necessarily the area with the most room or even inside the ‘blob’ (the geometric center of a ‘C’ shaped blob is actually outside the shape). In order to solve this problem, each blob is processed one at a time and the coordinates with the largest, minimum distance to an edge is returned. This is the place inside of a blob where the shortest distance in any direction to the edge is larger than the shortest distance from any other place.

1. **Combining contours and labels**

Finally, all contours (found in step 3) are drawn onto a blank canvas and numbers are written in at the optimal locations (found in step 4).

***Language:*** *Python*

***Libraries:*** *OpenCV, PIL, NumPy, Matplotlib, Tkinter (for GUI)*