CS 545 - Homework 5

Written Portion – Daniel Miller

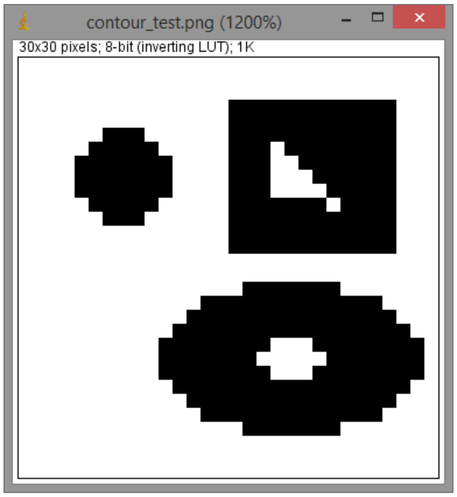
PROBLEM 1

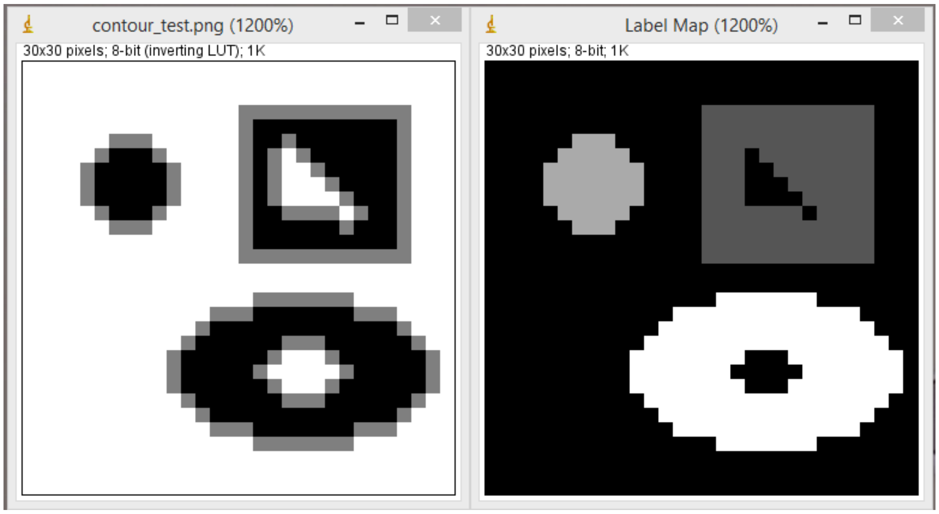
Chain Codes

In the included *Chain_codes.java* class, a simultaneous region labeling and contour tracing algorithm is implemented. Using the pseudocode described in Burger & Burge, this algorithm enumerates a list of all inner and outer contours in an image, while also generating a labeled image.

Also included is a *Contour.java* class which is used to handle the storage and comparison of the individual contours, as well as the generation of BOTH absolute and differential chain codes. Also, the getShapeNumber() function is used to generate Shape Numbers for each contour. These shape numbers are used in the hashCode() function for this class, allowing the main program to keep a Set of unique contours only.

The following images show the original test image, the detected contours, and the newly labeled regions.

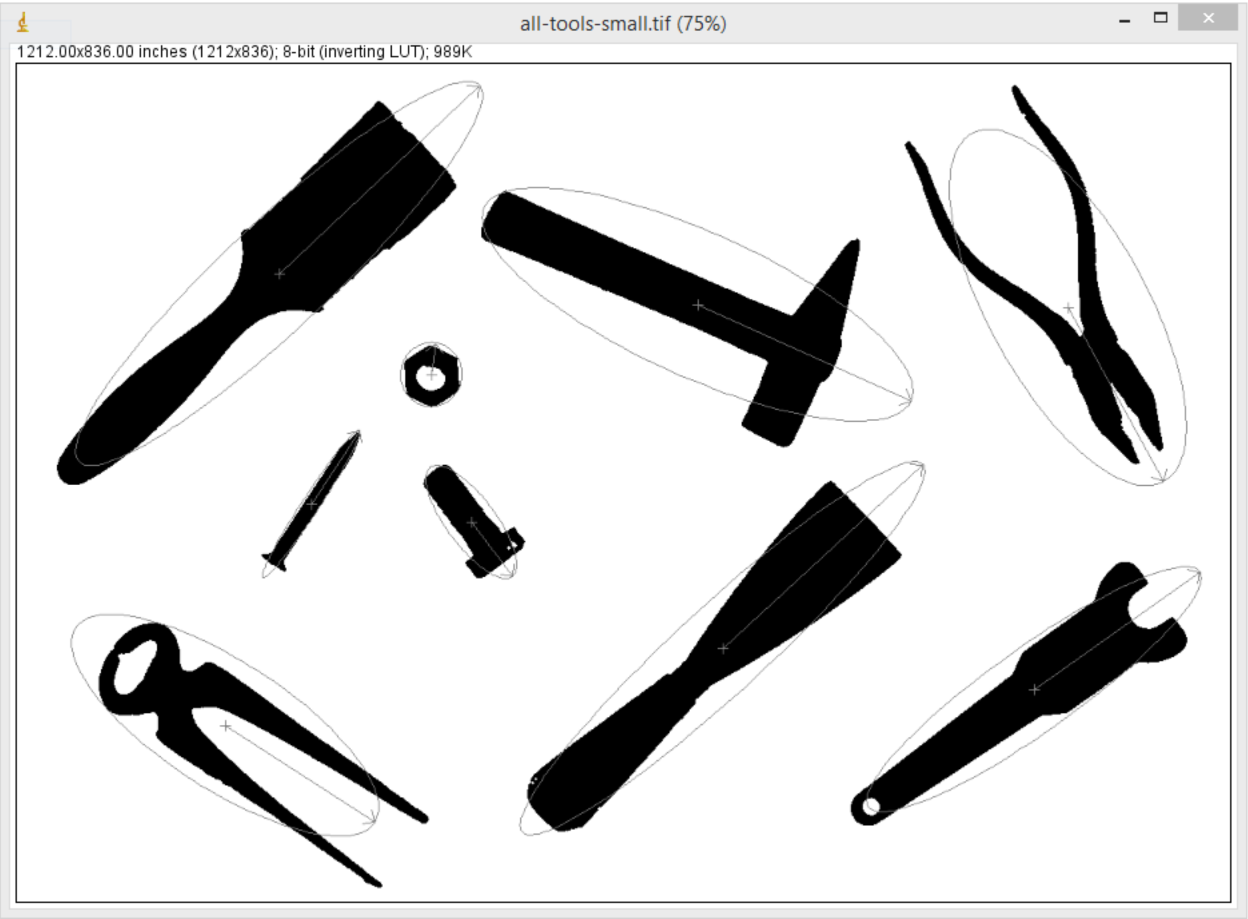




PROBLEM 2

Region Labeling

The *Region_labeling.java* class labels each binary region, calculates the orientation and eccentricity for each, and then displays them as a vector and ellipse over the original image. The result of applying this filter to the book's "all-tools-small.tif" image, yielding nearly identical results:


all-tools-small.tif (75%)
1212.00x836.00 inches (1212x836); 8-bit (inverting LUT); 989K

The logic and processing for this function is completely independent of the code used for Problem 1, and is based on a depth-first flood fill, which utilizes an "expanded pixels" image in memory to ensure the stack remains as small as possible. The various drawing methods are implemented in *Utils.java*, and the Region Labeling utilities, such as orientation and eccentricity calculation, are implemented as static methods in the *Region_Utils.java* class.

PROBLEM 3

Convert **RGB** (0, 184, 160) to **HSV**

We begin by calculating the Saturation (S_{HSV}) as follows:

$$S_{HSV} = \begin{cases} \dfrac{C_{rng}}{C_{high}} & for\ C_{high} > 0 \\ 0 & otherwise \end{cases}$$

$$C_{max} = 255 \quad C_{high} = \max(R, G, B) \quad C_{low} = \min(R, G, B) \quad C_{rng} = C_{high} - C_{low}$$

$$\therefore \quad C_{high} = 184 \qquad C_{low} = 0 \qquad C_{rng} = 184 - 0 = 184$$

$$\boxed{S_{HSV} = \frac{C_{rng}}{C_{high}} = \frac{184}{184} = 1.000}$$

Next, the Luminance/Value is calculated using the terms above:

$$\boxed{V_{HSV} = \frac{C_{high}}{C_{max}} = \frac{184}{255} \cong 0.722}$$

Finally, the Hue is calculated as shown here:

$$R' = \frac{C_{high} - R}{C_{rng}} = \frac{184 - 0}{184} = 1.000$$

$$G' = \frac{C_{high} - G}{C_{rng}} = \frac{184 - 184}{184} = 0.000$$

$$B' = \frac{C_{high} - B}{C_{rng}} = \frac{184 - 160}{184} = \frac{24}{184} = 0.1304$$

$$H' = \begin{cases} B' - G' & \text{if } R = C_{high} \\ R' - B' + 2 & \text{if } G = C_{high} \\ G' - R' + 4 & \text{if } B = C_{high} \end{cases} \rightarrow 1.000 - 0.1304 + 2 = 2.8696$$

$$\boxed{H_{HSV} = \frac{1}{6} \cdot \begin{cases} H' + 6 & \text{for } H' < 0 \\ H' & \text{otherwise.} \end{cases} \rightarrow \frac{1}{6} \cdot 2.8696 \cong 0.4783}$$

Which yields the final answer:

$$\text{RGB}(0, 184, 160) = \boxed{\text{HSV}(0.4783, 1.0, 0.722)}$$

PROBLEM 4

Convert **HSV** (0.67, 0.7, 0.7) to **RGB**

The conversion from HSV to RGB is calculated as follows:

$$H' = (6 \cdot H_{HSV}) \bmod 6 = 4.02$$

$$\begin{aligned} c_1 &= \lfloor H' \rfloor = 4 & x &= (1 - S_{HSV}) \cdot V_{HSV} = 0.21 \\ c_2 &= H' - c_1 = 0.02 & y &= (1 - (S_{HSV} \cdot c_2)) \cdot V_{HSV} = 0.6902 \\ & & z &= (1 - (S_{HSV} \cdot (1 - c_2))) \cdot V_{HSV} = 0.2198 \end{aligned}$$

$$(R', G', B') = \begin{cases} (V_{HSV}, z, x) & \text{if } c_1 = 0 \\ (y, V_{HSV}, x) & \text{if } c_1 = 1 \\ (x, V_{HSV}, z) & \text{if } c_1 = 2 \\ (x, y, V_{HSV}) & \text{if } c_1 = 3 \\ (z, x, V_{HSV}) & \text{if } c_1 = 4 \\ (V_{HSV}, x, y) & \text{if } c_1 = 5 \end{cases} = (0.2198, 0.21, 0.7)$$

The (R’, G’, B’) tuple is then normalized to the range [0, 255]:

$$\begin{aligned} R &= \min(\text{round}(255 \cdot R'), 255) = 56 \\ G &= \min(\text{round}(255 \cdot G'), 255) = 54 \\ B &= \min(\text{round}(255 \cdot B'), 255) = 178 \end{aligned}$$

Which yields the final answer:

$$\text{HSV}(0.67, 0.7, 0.7) = \boxed{\text{RGB}(56, 54, 178)}$$

PROBLEM 5

Convert **RGB** (124, 220, 0) to **YIQ**

The R, G, B components are first normalized to the [0, 1] range:

$$R' = \frac{124}{255} = 0.4863 \qquad G' = \frac{220}{255} = 0.8627 \qquad B' = \frac{0}{255} = 0.0$$

The Luminance (Y) component is then computed, which is shared with the YUV color space.

$$Y = 0.299 * R' \ + \ 0.587 * G' \ + \ 0.114 * B'$$

$$Y = 0.6518$$

$$U = 0.492 \cdot (B' - Y) = -0.3207 \qquad V = 0.877 \cdot (R' - Y) = -0.1452$$

$$\begin{pmatrix} I \\ Q \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} \cos\beta & \sin\beta \\ -\sin\beta & \cos\beta \end{pmatrix} \cdot \begin{pmatrix} U \\ V \end{pmatrix}$$

$$\begin{pmatrix} I \\ Q \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0.8387 & 0.5446 \\ -0.5446 & 0.8387 \end{pmatrix} \cdot \begin{pmatrix} -0.3207 \\ -0.1452 \end{pmatrix} = \begin{pmatrix} 0.0526 \\ -0.3481 \end{pmatrix}$$

Which yields the final answer:

$$\mathrm{RGB}(124, 220, 0) = \boxed{\mathrm{YIQ}(0.6518, 0.0526, -0.3481)}$$

PROBLEM 6

Convert **YIQ** (1.0, 0.3, 0.3) to **RGB**

The matrix form of the previous conversion (RGB to YIQ) is shown here:

$$\begin{pmatrix} Y \\ I \\ Q \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{pmatrix} \cdot \begin{pmatrix} R' \\ G' \\ B' \end{pmatrix}$$

The inverse transformation can be calculated using simple matrix math, as shown here:

$$\begin{pmatrix} R' \\ G' \\ B' \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{pmatrix}^{-1} \cdot \begin{pmatrix} Y \\ I \\ Q \end{pmatrix}$$

$$\begin{pmatrix} R' \\ G' \\ B' \end{pmatrix} = \begin{pmatrix} 1.000 & 0.9557 & 0.6199 \\ 1.000 & -0.2716 & -0.6499 \\ 1.000 & -1.1082 & 1.7051 \end{pmatrix} \cdot \begin{pmatrix} 1.0 \\ 0.3 \\ 0.3 \end{pmatrix} = \begin{pmatrix} 1.4727 \\ 0.7245 \\ 1.1791 \end{pmatrix}$$

The results are then scaled down to the [0, 255] range, as shown in the last step of Problem 4. The results show that this conversion has saturated the red and green channels, which are clamped to a maximum value of 255. This yields the final results as:

$$\mathrm{YIQ}(1.0, 0.3, 0.3) = \boxed{\mathrm{RGB}(255, 185, 255)}$$