1. Processing of Color Images

One often desires to process color images in order to locate features of a given color. For this reason the different color models are appropriate. It is important to maintain hue since the human vision system is very sensitive to changes in hue. If one has a blue color for water then changing the value or luminance would brighten or darken the color, and changing the saturation would change the color from a light blue to a dark blue. Both of these changes are not overly dramatic but changing the hue from blue to red would greatly affect our perception of the scene. Most often the data are collected from an RGB devices. Processing images with the RGB models makes it difficult to maintain hue and makes the RGB model not desirable in some cases [Weeks, 1996, pp.257]. One can process an RGB image with linear filters because hue is preserved with linear filters [Weeks, 1996, pp. 259, 279]. This would be the case for linear noise reducing filters such as convolution filters. One should note that noise that is uncorrelated between the RGB channels can affect the hue and therefore the perception of the image. One should note that RGB is dependent upon the imaging system [Haeghen, Naeyaert, Lemahieu, and Philips, 2000]. The CIE $L^*a^*b^*$ perceptually uniform color space is often used. However, the RGB transform to $L^*a^*b^*$ space may not come from a well defined RGB space but rather a device dependent one. Therefore, the $L^*a^*b^*$ color may not be valid or have the perceptually uniform properties. In this section some common processing methods are discussed.

1.1 Color Enhancement

One can enhance the color perception by increasing the saturation of a HSV image. This will make the colors less pale and more distinct [Weeks, 1996, pp.276]. The following figure shows an image where the saturation has been increased. The greens and reds are more distinct.

a. Original image **b. Enhanced image**

(original image from web site www.cs.washington.edu/imagedatabase/groundtruth, link provided by CMU computer vision test images web site)

Figure 1. Color enhanced image

1.2 Color smoothing.

One can apply a smoothing template to a color image. If one has an RGB image then one may smooth the individual components. One may convert to a HSV image and then smooth on the value (V) component [Gonzalez and Woods, 2002, pp. 328].

1.3 Color Sharpening

One may sharpen a color image in a manner similar to sharpening a gray-scale image. One may sharpen each component of an RGB image or convert to HSV and sharpen the value component [Gonzalez and Woods, 2002, pp. 330]. Typical sharpening template filters are given below. The origin is at the center of the template and is indicated with the circle.

-1	-1	-1
-1	9	-1
-1	-1	-1

Figure 2. Edge Enhancement Filter

The following figure shows the filter applied to each band of an RGB image.

a. Original image **b. Sharpened RGB image**

Figure 3. Sharpened image

The following figure shows the enhancement filter applied to the value band of the HSV image.

a. Original image **b. Enhance HSV image**

Figure 4. Sharpened HSV image

1.4 Color Segmentation and distance measures.

Color slicing refers to dividing the image into parts or segmenting the image according to each pixel's closeness to some target color [Gonzalez and Woods, 2002, pp. 320]. The following figure demonstrates the method. The original image is converted to the HSV format. A point is sampled from the dress as p_blue = [.56 .56 .46]. In the output image a point is labeled with a one iff (tia(1) – p_blue(1) < b) & (tia(2) – p_blue(2) < b) where b is a closeness parameter.

a. Original image **b. Output b=.04** **c. Output b=.08**

Figure 5. Image processed with closeness algorithm

Color distance measures can be used to segment an image [Gonzalez and Woods, 2002, pp. 233]. If one has a target color vector that one desires to use for segmenting the image then one can segment it by finding colors close to the target color. Let the target color be u. Then a point is close to u if $d(g(p),u) \leq b$ where b is a parameter and d is a suitable distance function. A simple distance function is Euclidian distance given by $d(g(p),u) = [(g(p)\text{-}u)^t * (g(p)\text{-}u)]^{\frac{1}{2}}$ considering the color vectors as a column matrix and performing matrix multiplication. The following figure shows this method applied. The target color vector was selected from the chin. The image was in HSV format and the HS components were used in the segmentation.

a. Image of Tia b. Segmented image

Figure 6. Segmented image

The covariance matrix is defined in the following matter. Let z by the color vector. Let μ be the average color vector. Then the covariance matrix is $C=E\{(z-\mu)^2\}$ is the expected value of $(z-\mu)^2$. The covariance matrix can be estimated in the following manner. Let $U=\{[z(p)] | p \in W\}$ be formed by putting all the color vectors in window W in the rows of matrix U. Let m by the mean vector matrix formed by putting the μ vector in the columns of m. Then $C=\frac{1}{N}[U-m]^t[U-m]$ where N is the number of points in the window. For an RGB image C will be a 3 by 3 matrix. The distance function is then modified to be $d(g(p),u)=[(g(p)-u)^t * C^{-1} * (g(p)-u)]^{\frac{1}{2}}$ where u is the target color [Gonzalez and Woods, 2002, pp. 233].

The following figure shows this approach on an RGB image. The ranges on RGB values were from 0 to 255. The threshold was 6. The target color was taken from the skin area. The covariance matrix and its inverse are given below. The distance selected was 6.

C

3149	2704	2485
2704	2788	2728
2485	2728	2746

C^{-1}

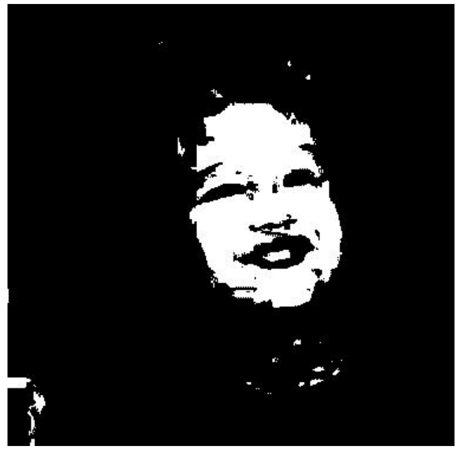
.005492	-.01681	.01174
-.01682	.06455	-.04892
.01174	-.04892	.03835

a. Original image **b. Segmented image**

Figure 7. Segmented image using covariance matrix

The next figure has the distance from the target pixel reduced to 3. A smaller region results.

a. Original image



b. Segmented image

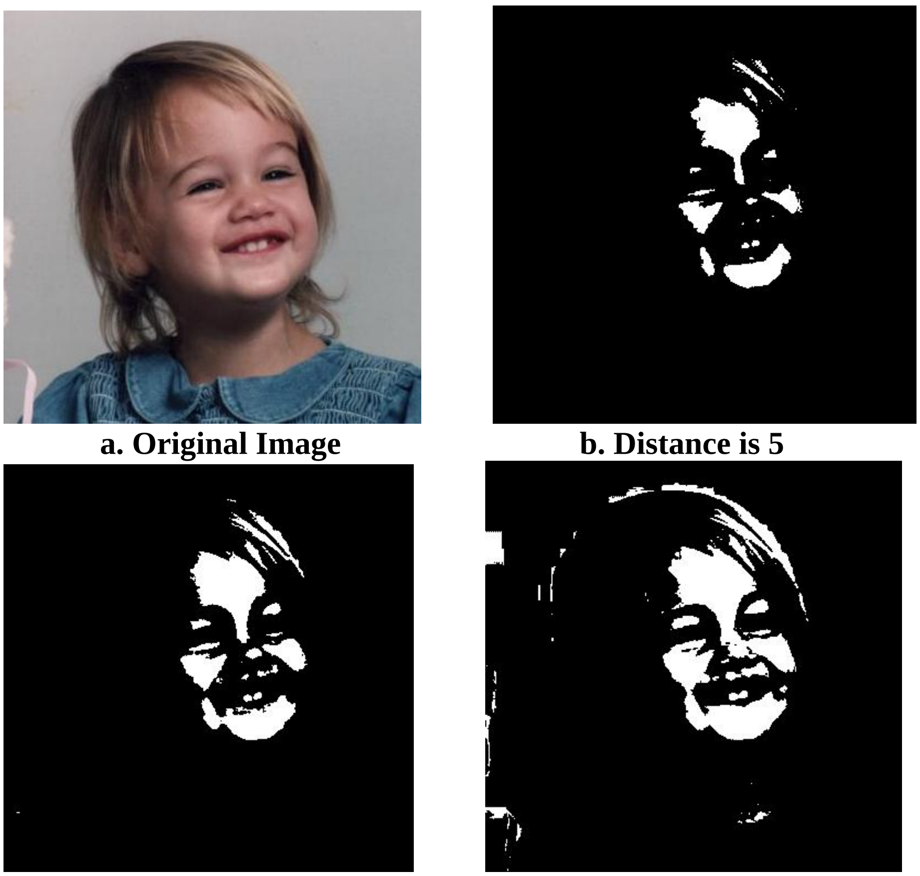
Figure 8. Segmented image using covariance matrix

One may be interested in segmenting the image and leaving the rest of the image as a gray-scale image [Weeks, 1996, pp.278]. One can accomplish this in the HSV color model by determining the points in the image that are associated with the object and setting the saturation to zero for the rest of the points. The following image shows this processing. The target object was the blue part of the image. The saturation of the points in the located region has been slightly increased while the value component is unchanged. If one processed the image in RGB color format one would have to set the RGB values of the points not in the target object so that the RGB values are all equal to get the gray color for the points that are not blue.

a. Original Image **b. Segmented image**

Figure 9. Segmented Image

One can also process the data in the L*a*b* color model. This model has the property that changes in $\Delta E^*_{1,2}=\left[\left(L^*_1\text{-}L^*_2\right)^2+\left(a^*_1\text{-}a^*_2\right)^2+\left(b^*_1\text{-}b^*_2\right)^2\right]^{\frac{1}{2}}$ between two points is related to the perceptual distinction of the two points. In this method we compute the distance between a target point and the other points in the image. If the distance is close then the point is in the target region. The following image shows the application of this method. The distance between the point and the target point varies from 5 to 9.



a. Original Image **b. Distance is 5**

c. Distance is 7 **d. Distance is 9**

Figure 10. Segmentation in L*a*b* Color Space

1.5 Color Gradient

One can compute the color gradient by computing it on each component of an RGB image [Gonzalez and Woods, 2002, pp. 335]. A typical template would be the Sobel correlation templates to compute g_x and g_y where g is the RGB image function.

1/8

-1	0	1
-2	0	2
-1	0	1

g_x

1/8

1	2	1
0	0	0
-1	-2	-1

g_y

The overall gradient would by $\nabla g=(g_x, g_y)$. The magnitude would be $\|\nabla g\|=\left(g_x^2+g_y^2\right)^{\frac{1}{2}}$. The angle would be $\theta=\tan^{-1}\left(\frac{g_y}{g_x}\right)$. The magnitude and angle are computed for each component of the RGB image separately. The following figure shows an example of the processing.

a. original image

b. red gradient image **b. green gradient image** **c. blue gradient image**

Figure 11. RGB gradient images

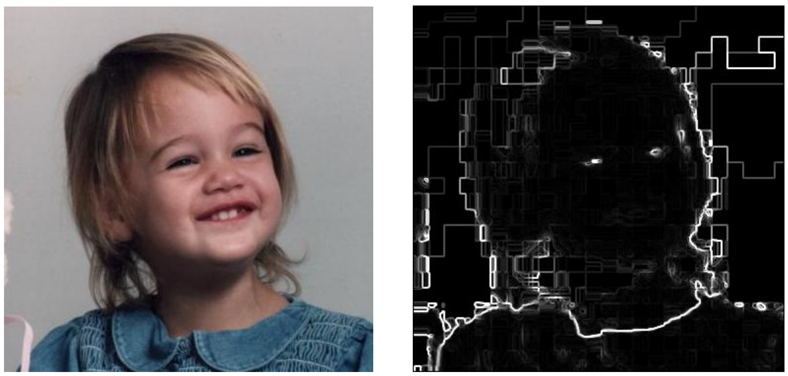
Another edge function can be defined as follows [Gonzalez and Woods, 2002, pp. 337]. Let $u=(R_x,G_x,B_x)$ and $v=(R_y,G_y,B_y)$. Then $g_x^2=\langle u,u\rangle=R_x^2+G_x^2+B_x^2$, $g_y^2=\langle v,v\rangle=R_y^2+G_y^2+B_y^2$ $g_{xy}=\langle u,v\rangle=R_xR_y+G_xG_y+B_xB_y$. The direction of maximum change is $\theta=\frac{1}{2}\tan^{-1}*\left(\frac{2g_{xy}}{g_{xx}g_{yy}}\right)$. The amount of change in direction θ is given by

$r=\left[\frac{1}{2}\left(g_{xx}+g_{yy}\right)+\left(g_{xx}-g_{yy}\right)\cos(2\theta)+2g_{xy}\sin(2\theta)\right]^{\frac{1}{2}}$. The following figure shows the output of this edge detection process.

a. Original Image **b. Output of edge operator**

Figure 12. Image and output of edge operator

The human eye is sensitive to changes in hue. This indicates that an edge detector that focuses upon the hue component may be useful [Weeks,1996 pp. 284]. An edge detector that operates upon the RGB components detects edges from changes in saturation and value as well as hue. With hue one must allow for the fact that hue varies modulus 360 degrees. In other words 0 and 360 degrees are adjacent. One way to do this is to use the sin and cos functions. Let $C(\theta) = \cos(\theta)$ and $S(\theta)= \cos(\theta)$ for the hue angle. Let $r(\theta)=\|\nabla C(\theta)\|+\|\nabla S(\theta)\|$. Here r is the sum of the magnitudes of the gradients of C and S. These gradients quantities can be measured using the methods previously described. An example is shown in the following image.



a. Original Image **b. Hue edge detection**

Figure 13. Hue Edge Detection

One can also process the data in the L*a*b* color model. This model has the property that changes in $\Delta E^*_{ab} = \left[\left(L^*_1 - L^*_2\right)^2 + \left(a^*_1 - a^*_2\right)^2 + \left(b^*_1 - b^*_2\right)^2 \right]^{\frac{1}{2}}$ between two points is related to the perceptual distinction of the two points. In this method we compute the gradient of each of the components of the image in L*a*b* and use the magnitude of the gradient image as the edge output.

a. Original Image **b. L*a*b* edge detection**

Figure 14. L*a*b* Edge Detection