Project 1 - Color, Texture, and Shape

Max Ehrlich, Paul Ton, Xiaoran Liu, Cheng Liu, Miao Li

* Color Histogram
* Color Correlogram
* Histogram of Oriented Gradients
* Local Binary Pattern

We used the very basic pattern of LBP algorithm that only calculate the center pixel with 8 pixels that around it. And we also create a structure named LBP including the bin id and the value.

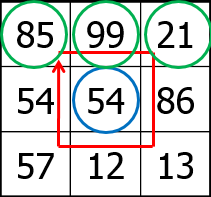
**Process:**

1. Load the image, convert the pixel’s RGB to grayscale:

Grayscale = (R\*299 + G\*587 + B\*114 + 500)/1000;

The reason why I didn’t use float value is that float calculation may decrease the speed of the program.

1. Compare with pixels around it, generate the binary pattern code. If the grayscale value of adjacent pixel is no smaller than that in the middle, we set as ‘1’, otherwise we set as ‘0’. The algorithm followed a sequence started from top left of the matrix. As image below:



1. Exclude the same LBP and convert binary to decimal.

I do this by transfer every digit of the binary pattern until the last element is ‘1’;

1. Calculate each LBP’s number count.

Scan the image for every pixel, if the outcome exist in our vector, increase the count value of that bin by 1, otherwise insert a new LBP value in the vector.

As the max value of an eight-digit binary is 255, we set the vector size as 256 at the very beginning, if we get a new pattern, we insert it into the vector at the position that equals the bin id, so that we don’t need to do the sorting after calculation.

1. Return a vector output.

**Output:** LBP histogram vector.

* L1 Distance
* L2 Distance
* Chi – Square
* Cosine Similarity
* Histogram Intersection