Digital Image Processing

Histogram and Histogram processing

Histogram

- Simply counting how many times a value is encountered within the image. Generally displayed as a bar chart.
- Counting does not need to be done on countable integer values, it is possible to create bins for a range of values. For instance, in a floating point image, each bin could have a range of 0.1. In this case, there will be 10 bins, each bin will collect the values falling into its range.
- It is a great tool to analyze image composition as well as any problems with it
- There are transformations involved in histograms to improve the image
- As you go right in the histogram values get lighter

Types

- Number of occurrences: Simply find the counts of every value in the image. Sum of all entries should be the number of pixels.
- Probability distribution function (PDF): Calculate the probability of finding a value in the image. Divide count by the number of pixels. Sum of all entries should be 1.
- Cumulative distribution function (CDF): Sum of probabilities until a specific value. Sum all probabilities to that point. The last entry should be 1.

Operations

- Brightness (b): adjusts the brightness of the image. Generally brightness is pair by contrast operation as brightness may decrease the overall contrast in the image. May cause overflow or underflow in the value. You must handle these cases.

  $O_{x,y} = I_{x,y} + b$
- Contrast (c): Increases (c > 1) or decreases (c < 1) the contrast of the image. The method works simply by pushing away or pulling away the values further from the central value.

  $O_{x,y} = c(I_{x,y} - 2^{(B-1)}) + 2^{(B-1)}$
- Gamma (γ): Adjusts the brightness of the image in a more realistic way. Compared to brightness operation, this method is costly in terms of processing. Does not cause overflow or underflow.

  $O_{x,y} = (2^B - 1)\left(\frac{I_{x,y}}{2^B-1}^{\gamma}\right)$
- Inverse: Inverts the values in the image, creating a negative image.

  $O_{x,y} = (2^B - 1) - I_{x,y}$
- Contrast stretching: expands the dynamic range of the image (and the contrast through that) by ensuring minimum value in the image is 0 and the maximum value corresponds to the maximum allowed by the image type. B is the number of bits in the image. **minarg** and **maxarg** finds the minimum and maximum value in the image. May not be effective in some images.

  $min = \textbf{minarg} I_{x,y}$

  $max = \textbf{maxarg} I_{x,y}$

  $O_{x,y} = (I_{x,y} - min)\frac{2^B-1}{max-min}$
- Histogram equalization (histeq): Increases the dynamic range of the image as well as improves the distribution of intensities. Works on all types of images. As with other contrast enhancement methods, this method will make the noise in the image more noticeable. In order to perform histeq, follow these steps:
- Calculate $2^B - 1$, this is the highest intensity possible. 255 in regular images
- Calculate counts of every intensity, you will need an array of size 256 call it *histogram*. Don't forget to initialize your array to 0. You could do that using $int\ histogram[256] = \{\}$;
- You will do these steps per intensity. Thus, you will loop from 0 to 255.
- Calculate PDF, simply divide the counts of intensities ($histogram$) by the number of pixels ($width \times height$). *In programming don't forget to convert you integer numbers to float for this operation.*
- Calculate CDF, create another array for this. Keep current sum starting from 0. At each step, add the PDF of that intensity to your count and set it to CDF array.
- Multiply CDF with highest intensity. This will give us *replacement* intensities
- Finally, for each pixel, replace the pixel value by the value stored in *replacement* for the current pixel value. ie $replacement[image(x, y)]$