Programming Assignment 1

CS 485  
Shubham Gogna  
Due: February 19, 2015  
Dr. Mircea Nicolescu

Part One: Gaussian Smoothing

For (a), as sigma increased, the values were smoothed out. What started as peaks of 0s and 1s became gradually increasing decimal values from 0 to 255 (when normalized between those values).

|  |  |
| --- | --- |
| Sigma | Mask Vector |
| 1 | 0.0544887  0.244201  0.40262  0.244201  0.0544887 |
| 5 | 0.00453456  0.00718308  0.0109324  0.0159862  0.0224598  0.0303176  0.0393198  0.0489955  0.0586583  0.0674731  0.0745693  0.0791804  0.0807799  0.0791804  0.0745693  0.0674731  0.0586583  0.0489955  0.0393198  0.0303176  0.0224598  0.0159862  0.0109324  0.00718308  0.00453456 |
| 11 | 0.00180578  0.00224791  0.00277526  0.00339812  0.00412653  0.00496984  0.00593623  0.00703217  0.00826188  0.00962674  0.0111248  0.0127501  0.0144926  0.0163376  0.018266  0.0202539  0.0222733  0.0242924  0.0262766  0.0281888  0.0299913  0.0316465  0.0331182  0.034373  0.0353818  0.0361204  0.036571  0.0367224  0.036571  0.0361204  0.0353818  0.034373  0.0331182  0.0316465  0.0299913  0.0281888  0.0262766  0.0242924  0.0222733  0.0202539  0.018266  0.0163376  0.0144926  0.0127501  0.0111248  0.00962674  0.00826188  0.00703217  0.00593623  0.00496984  0.00412653  0.00339812  0.00277526  0.00224791  0.00180578 |

The graph below shows that as the sigma increased, the data smoothed out more. This is an expected behavior because the larger sigma takes a larger number range into account which creates a smooth curve out of sharp jumps in value.

Graph shows how the data is distributed after a Gaussian is convolved with the data.

For (b), the resulting distributions of data were very similar with slight visible exceptions for the single convolution. It is possible that it is the result of a multiplying with an irrational number which resulted in a non-integer sigma and mask size (around 7.07 for the sigma). The floating point precision may not have been enough to get the distributions to completely overlap, but the graph indicates that they were very close. This is to be expected since it is the property of two Gaussian convolutions.

Graph shows the very close similarity between the two convolution and the single convolution results

For (c), the 1D Gaussian mask was multiplied by its transpose to get a 2D Gaussian mask. That 2D mask was convolved with the image to get the results for (c). For (d), the transpose of the 1D mask was applied to every row and then the mask was applied to every column. Theoretically, the results are supposed to be the same, but the performance on the second method (d) should be better in terms of Big-O. The images below show comparisons between the two methods.

|  |  |  |
| --- | --- | --- |
| Sigma | 2D Mask Applied Once | 1D Mask Applied Twice |
| N/A | Original Image | |
| 1.0 | 2D Mask, Sigma = 1.0 | 1D Mask, Sigma = 1.0 |
| 5.0 | 2D Mask, Sigma = 5.0 | 1D Mask, Sigma = 5.0 |
| 11.0 | 2D Mask, Sigma = 11.0 | 1D Mask, Sigma = 11.0 |

Part Two: Edge Detection

In order to get the gradient image in the x direction, the Sobel operator with 0s horizontally was used (S\_x). To get the gradient image in the y direction, the Sobel operator with 0s vertically was used (S\_y). The gradient magnitude image was obtained by taking the square root of the sum of pixel values squared from each gradient image. The gradient direction image was obtained by taking the atan2 function between I\_y and I\_x. Unfortunately, the function returns a value between negative pi and positive pi which does not show up very well on an image. In order to show the data, the values were normalized between [0, 255].

Below are the corresponding images for part two.

|  |  |
| --- | --- |
| Vertical Sobel Operator applied to Lenna | Horizontal Sobel Operator applied to Lenna |
| Gradient Magnitude Image for Lenna | Gradient Direction Image for Lenna |
| Magnitude Image (Threshold = < 100 = 0 ) |  |
| Vertical Sobel Operator applied to SF | Horizontal Sobel Operator applied to SF |
| Gradient Magnitude Image for SF | Gradient Direction Image for SF |
| Magnitude Image (Threshold = < 100 = 0 ) |  |

Part Three: Gaussian Pyramid

The Gaussian Pyramid was implemented in a very simple manner. Start from an image size of 256 by 256, a main loop in the program repeats a process until the image size reaches 4 by 4. The process involves saving the current image, convolving with the Gaussian mask, and the downscaling the image by 2 in each dimension. So in one operation, the image shrinks by a factor of 4. The following are the generated images in the Gaussian Pyramid of Lenna. Since there are no intermediate images, the value of ‘s’ is 1.

|  |  |
| --- | --- |
| Size | Image |
| 256 x 256 |  |
| 128 x 128 |  |
| 64 x 64 |  |
| 32 x 32 |  |
| 16 x 16 |  |
| 8 x 8 |  |
| 4 x 4 |  |

As for the images in the pyramid, there is an obvious loss of detail because of the blurring and shrinking. If the images from the top of the pyramid were to be expanded to the size of the bottom, there would be a clear lack of descriptive detail. However, since there is an initial blur with a Gaussian, the detail is better than what it would be with averaging neighboring pixels and downsizing.