Machine Perception COMP3007

Image Processing

Exercise 1 - Colour Conversion

Write a program that reads a colour image and performs each of the following colour conversions

- Conversion from RGB to gray.
- Conversion from RGB to HSV.
- Conversion from RGB to Luv.
- Conversion from RGB to Lab.

Observe the result and note the effect of each conversion.

Exercise 2 - Linear Filtering

Using the function `filter2D` in OpenCV, write a program that loads a gray-scale image and performs linear filtering with the following kernels/filters

- Prewit kernels

$$\mathbf{K}_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}, \ \mathbf{K}_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

- Sobel kernels

$$\mathbf{K}_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \ \mathbf{K}_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

- Laplacian kernel

$$\mathbf{K}_{\text{Laplacian}} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

- Gaussian kernel with $\sigma = 1$

$$\mathbf{K}_{\text{Gaussian}} = \frac{1}{273} \begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix}$$

Note that `filter2D` only computes the correlation, and thus you will need to flip the kernel to do actual convolution. Observe the result and comment on the effect of each filter.

There is also a function `GaussianBlur` in OpenCV that implements a more general Gaussian filter. Experiment with this filter for different values of the standard deviation σ_X and comment on the result when σ_X is varied from small to large.

Exercise 3 - Median Filtering

In this exercise, you are given a gray-scale image that has been corrupted with noise. Write a program that performs median filtering of this corrupted image. Observe how the median filter helps remove the corruptions.

Exercise 4 - Pixel Transform

In this exerise, you are asked to write a program that peforms affine transformation to all pixels of a gray-scale image that suffers from poor contrast and brightness. Compare the results and observe how adjusting α and β helps improve the contrast and brightness of the image.

Exercise 5 - Histogram Equalization

Using OpenCV's histogram equalization method `equalizeHist`, write a program that improves the contrast of an input image. Visually inspect the output image and comment how histogram equalization changes its contrast. Compute the histograms of the input and equalized images and comment how the histogram has been stretched out more widely and uniformly.

Exercise 6 - Morphology

In this exercise, you will familiarize yourself with two morphological opearations, namely dilation and erosion. Using OpenCV's functions `erode` and `dilate`, write a program to perform either dialation or erosion to the input images. Use `getStructuringElement` to obtain the structuring elements of these operators. Vary the dilation/erosion size and observe how it affects the result.

Advanced morphological operations can be constructed based on erosion and dilation. In the second part of this exercise, you will investigate the following operations

- **Opening**: this is useful for removing small objects, and is defined as

  `open`(src,element) = `dilate`(`erode`(src,element))

- **Closing**: this is useful for removing holes, and is defined as

  `close`(src,element) = `erode`(`dilate`(src,element))

- **Mophological gradient**: this is useful for finding the outline of an object, and is defined as

  `morph_gradient`(src,element) = `dilate`(src,element)-`erode`(src,element)

- **Blackhat**: which is the difference between the closing and its input image `blackhat`(src,element) = `close`(src,element) - src

End of Practical.