OSP Assignment 4 Readme.txt

1. **Salt and Pepper Noise Removal**

* Purpose of the code

“salt\_and\_pepper.cpp” implements generating an image with the salt-and-pepper noise and then removing the salt-and-pepper noise with median filtering from grayscale input and color input image.

* Environment

visual studio 2019 with opencv 2.4.13.6

* How to run this code

cl salt\_and\_pepper.cpp

* How to adjust parameters

to change input image: edit "input.jpg"

Mat input = imread("input.jpg", CV\_LOAD\_IMAGE\_COLOR);

to change the density of salt and pepper noise: edit parameter in Add\_salt\_pepper\_Noise function

Mat noise\_Gray = Add\_salt\_pepper\_Noise(input\_gray, 0.1f, 0.1f);

to change the method of boundary processing: edit parameter in Salt\_and\_pepper\_noise\_removal function

Mat Denoised = Salt\_pepper\_noise\_removal(noise, window\_radius, "adjustkernel");

1. **Gaussian noise removal**

* Purpose of the code

“Gaussian.cpp” code is for generating an image with the gaussian noise and remove the gaussian noise with two ways for grayscale and color input image. One is the gaussian filtering and the other one is the bilateral filtering.

* Environment

visual studio 2019 with opencv 2.4.13.6

* How to run this code

cl Gaussian.cpp

* How to adjust parameters

to change input image: edit "input.jpg"

Mat input = imread("input.jpg", CV\_LOAD\_IMAGE\_COLOR);

to change the method of boundary processing: edit the last parameter in gaussianfilter/bilateralfilter function

Mat Denoised = Gaussianfilter(noise, 3, 10, 10, "zero-padding");

Mat Denoised\_BF = Bilateralfilter(noise, 3, 10, 10, 0.15, "zero-padding");

1. **Adaptive thresholding using moving averages**

* Purpose of the code

“adaptivethresholdSkeleton.cpp” code implements adaptive thresholding with moving averages for grayscale input image. At each kernel the mean of intensity value gets to be a threshold for (i, j) pixel. The output is segmented by white and black.

* Environment

visual studio 2019 with opencv 2.4.13.6

* How to run this code

cl adaptivethresholdSkeleton.cpp

* How to adjust parameters

to change input image: edit "input.jpg"

Mat input = imread("input.jpg", CV\_LOAD\_IMAGE\_COLOR);

to change kernel size: edit “n” in adaptive\_thres function

output = adaptive\_thres(input\_gray, n, 0.9);

1. **K-means clustering**

* Purpose of the code

“kmeansSkeleton.cpp” is for image segmentation using k-means clustering. In this code, pixels are grouped based on intensity and position similarity. As the output of this code grayscale and color input image are segmented by the number of k centers.

* Environment

visual studio 2019 with opencv 2.4.13.6

* How to run this code

cl kmeansSkeleton.cpp

* How to adjust parameters

to change input image: edit "input.jpg"

Mat input = imread("input.jpg", CV\_LOAD\_IMAGE\_COLOR);

to change the number of centers: edit “clusterCount”

int clusterCount = 10;

to change trial number to finding appropriate centers

int attempts = 5;