**2020 Multimedia&Labs HW#2\_2**

**Due Date: 2020. 05. 05**

**ID/Name: 201533661 이승수**

**Lab3-2: Gaussian smoothing filtering**

**Source Code:**

//#include <opencv2/core/core.hpp>

//#include <opencv2/imgcodecs.hpp>

//#include <opencv2/highgui/highgui.hpp>

#include <opencv2/opencv.hpp>

#include <iostream>

#include <string>

using namespace cv;

using namespace std;

double alpha;

int beta;

int main(int argc, char\*\* argv) {

//Lab3-2.2: Gaussian smooth filtering

string src\_Path = "D:\\repos\_VS\\Project\_sourceIMG\\";//image source file path(folder)

//Load image

if (argc > 1) {

src\_Path = argv[1];

}

Mat src = imread((src\_Path + "lena\_noise.png").c\_str(), IMREAD\_GRAYSCALE);

//resize source image

Mat resized\_srcImage;

resize(src, resized\_srcImage, Size(256, 256));

//filtered Image

Mat new\_image = Mat::zeros(src.size(), src.type());

if (src.empty() || new\_image.empty()) {

cout << "Could not open or find the image" << std::endl;

return -1;

}

//insert kernel size

int ksize;

std::cout << "Enter the kernel size[3,5,7,25]: ";

std::cin >> ksize;

float sigma = 3.0;//sigma value

std::cout << "sigma: "<<sigma<<"\n";

//Gaussian Smoothing with k\*k size filter with sigma value

GaussianBlur(src,new\_image,Size(ksize,ksize),3.0);//sigma=1.5

//Display results

imshow("source Image", src);

imshow("gaussian filtered Image", new\_image);

//Wait until user exits program

std::cout << "program Ended. press any Key. ";

waitKey(0);

return 0;

waitKey(0);

return 0;

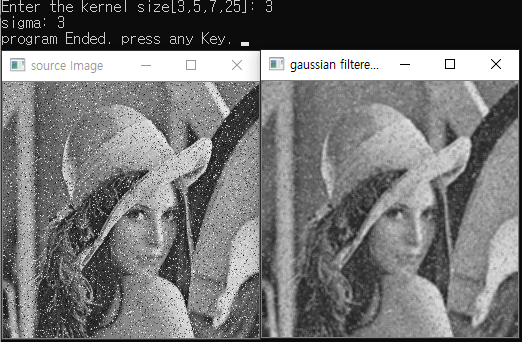
}

**Result:**

Case: ksize=3

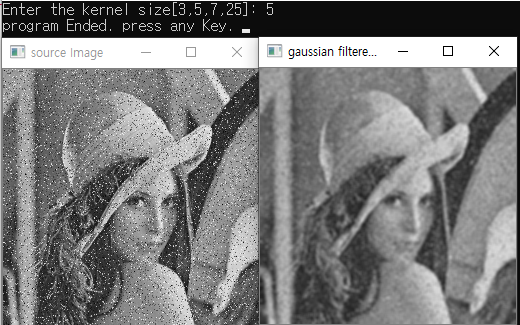
(sigma=1.5)





Case: ksize=5

(sigma=1.5)





Case: ksize=7

(sigma=1.5)





Case: ksize=25

(sigma=1.5)



