**2020 Multimedia&Labs HW#2**

**Dept: Software**

**ID/Name: 201533661 이승수**

**Lab3-2: Average 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;

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

//Lab3-2: average 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\_color.png").c\_str(), IMREAD\_GRAYSCALE);

//resize source image

Mat resized\_srcImage;

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

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;

}

//make kernel

int ksize;

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

std::cin >> ksize;

double temp;//filtered result

Mat kernel = Mat::ones(Size(ksize,ksize),CV\_32FC1);

for (int y = ksize / 2; y < src.rows - ksize / 2; y++) {

for (int x = ksize / 2; x < src.cols - ksize / 2; x++) {

//kerneling

temp = 0.0;

for (int i = 0; i < ksize; i++) {

for (int j = 0; j < ksize; j++) {

temp += src.at<uchar>(y + (i - ksize / 2), x + (j - ksize / 2)) \* kernel.at<float>(i, j);

}

}

temp = saturate\_cast<double>(temp / (ksize \* ksize));

new\_image.at<uchar>(y, x) = temp;

}

}

//Display results

imshow("source Image", src);

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

//Wait until user exits program

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

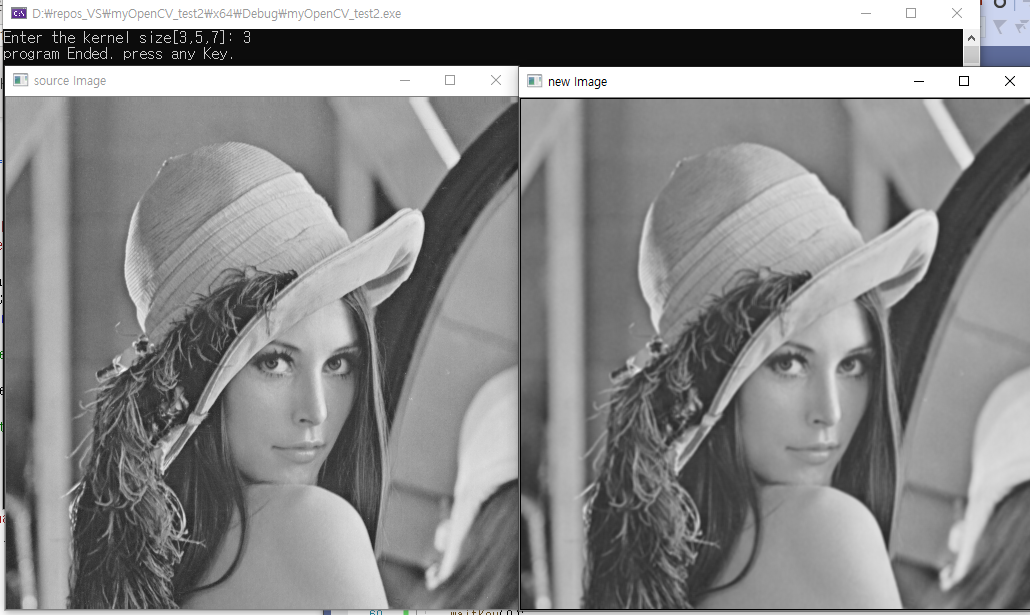
waitKey(0);

return 0;

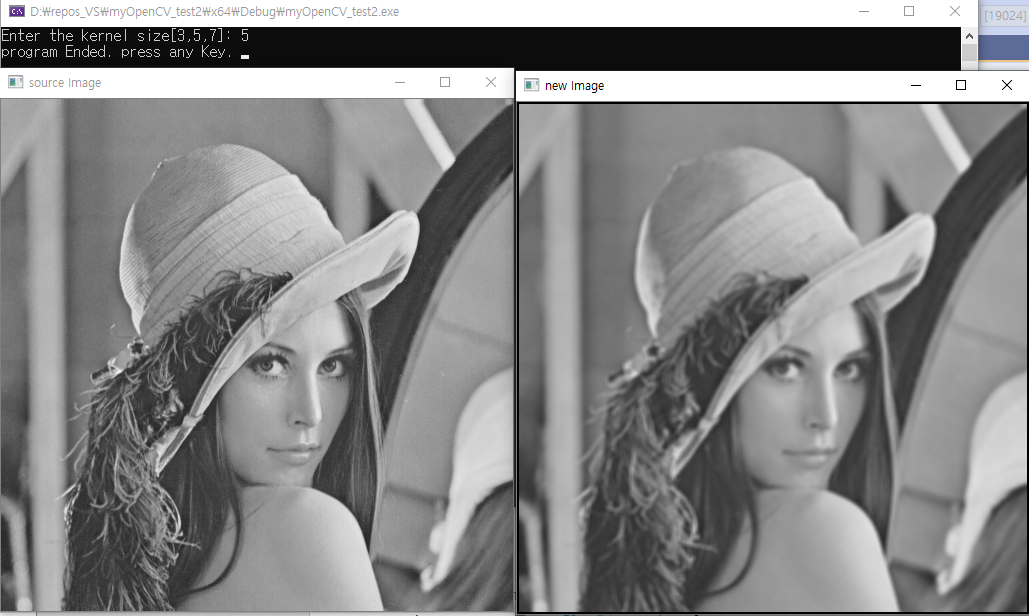
}

**Result:**

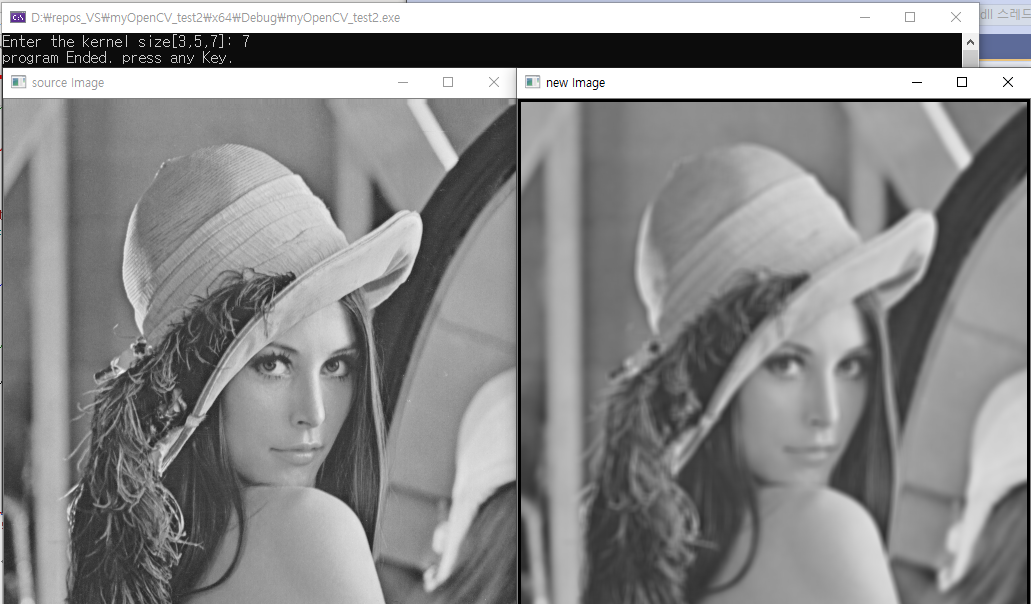
Case: ksize=3



Case: ksize=5



Case: ksize=7



Case: ksize=25

