KANGWON NATIONAL UNIVERSITY

컴퓨터비전 실습

실습3 | Filtering

실습기간 3/21 ~3/27 (PM 23:59)
실습과제 이루리 내 제출

CVMIPLAB @ KNU

과제 제출

- 문의방법
  - 컴퓨터비전 TA: 박진오, 이재훈 / 컴퓨터비전&의료영상처리연구실
  - 이루리 내 Q&A 게시판에 질문 작성

- 과제 제출 방법
  - **결과 이미지 파일과 main 포함된 cpp파일을 이루리 과제 제출란에 압축하여 제출합니다.**

실습 3-1 | Dissolve 구현

문제

주어진 두 "cat.bmp", "tibetfox.bmp" 파일을 읽고, 아래 수식에 따라 디졸브를 구현합니다.

$$f_{out}(j,i) = \alpha f_1(j,i) + (1-\alpha) f_2(j,i)$$

디졸브 시, $\alpha = 0.3$일 때와 $\alpha = 0.7$일 때를 각각 "dissolve_3.bmp", "dissolve_7.bmp"로 제출합니다.

※ 다음 슬라이드의 실행 결과를 보고 디졸브 순서에 유의하세요.

다만 영상과 같은 **실시간으로 움직이는 시퀀스를 구현하지 않아도 됩니다**

(파일 두 개만 생성하도록 해도 됩니다.)

요구 결과

결과 파일 "dissolve_3.bmp", "dissolve_7.bmp"과 main함수가 포함된 ".cpp" 코드 총 3개를 압축해 제출합니다.

실습 3-1 | Dissolve 구현

실행결과 동영상

실습 3-2 | Low Pass Filter 구현

문제

주어진 “salt_pepper.bmp” 파일을 읽고, Low Pass Filter를 구현하여 적용한 뒤 “salt_pepper_lpf.bmp”로 저장하세요.

Low pass filter 계수:

$$\begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix}$$

요구 결과

결과 파일 “salt_pepper_lpf.bmp”와 main함수가 포함된 “.cpp” 코드 두 개를 압축해 제출합니다.

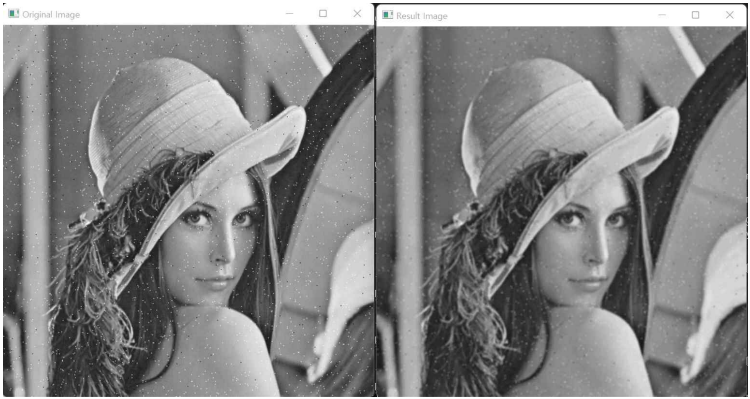
실행결과 및 코드

※ 필터를 적용할 때는 Element-wise로 곱하여 구현하고, **cv::filter2D를 사용하면 안 됩니다.**

※ 필터를 정의할 때는 일반 배열로 정의하는 것을 권장합니다.

실습 3-2 | Low Pass Filter 구현

실행결과



실습 3-3 | High Pass Filter 구현

문제

주어진 “tibetfox.bmp” 파일을 읽고, High Pass Filter를 구현하여 적용한 뒤 “tibetfox _hpf.bmp”로 저장하세요.

High pass filter 계수:

-1	-1	-1
-1	8	-1
-1	-1	-1

요구 결과

결과 파일 “tibetfox _hpf.bmp”와 main함수가 포함된 “.cpp” 코드 두 개를 압축해 제출합니다.

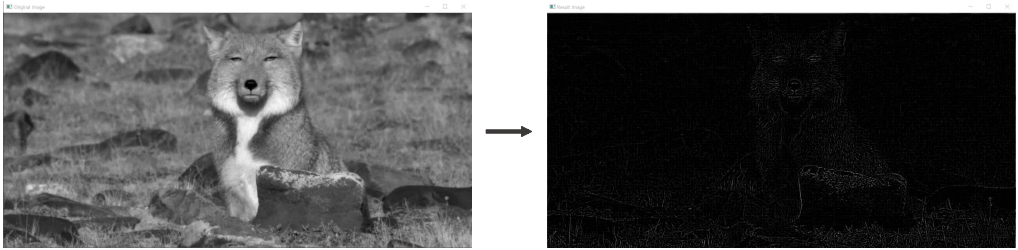
실행결과 및 코드

※ 필터를 적용할 때는 Element-wise로 곱하여 구현하고, **cv::filter2D를 사용하면 안 됩니다.**

※ 필터를 정의할 때는 일반 배열로 정의하는 것을 권장합니다.

실습 3-3 | High Pass Filter 구현

실행결과



Q&A

Q1. .bmp 이미지 파일 저장 방법

→ cv::imwrite("Result_Image_Name.bmp", result_image);

Q2. 초기 이미지 초기화

1. img.copyTo(result); → img를 비어있는 result에 복사
2. cv::Mat img; img.create(cv::Size(“row 크기”, “col 크기”), CV_8UC1);

※ CV_8UC1은 데이터 타입을 나타냄 (8U = 8비트이고 Unsigned형태, C1 = Channel이 1이다.)