

**과목명 | 디지털영상처리**

**실습# | 1**

**담당교수 | 정진우**

**학과 | 컴퓨터공학과**

**학년 | 3**

**학번 | 2010112406**

**이름 | 박도령**

**제출일 | 2016. 3. 31**

**실습에 사용한 이미지)**

**image.jpg / processed.jpg**

** **

**이미지 사이즈 통일을 위하여 저번 실습에 사용한 이미지(image.jpg)와 좌우 대칭으로 만들어 생성한 이미지(processed.jpg)를 사용하였다.**

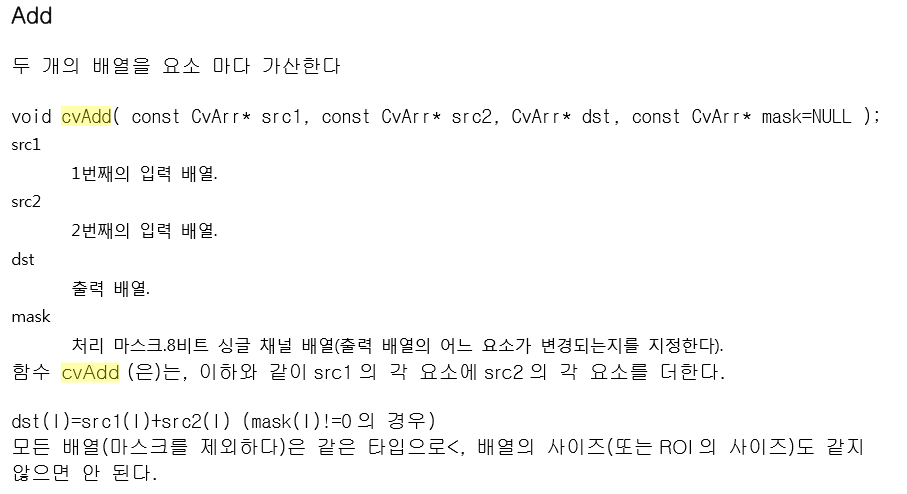
**실습 1)**

**(1) Addition with two images.(Input : any two images)**

♋ **동작 순서 및 원리**

0. 사이즈가 똑같은 이미지 2개를 생성한다.

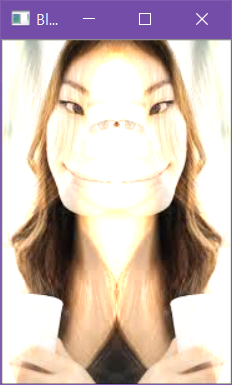
1. 이미지의 두 RGB값을 더하기 위해 (Clamping 처리 포함) Opencv에서 제공하는 함수 CvAdd를 사용한다. CvAdd의 내용은 다음과 같다.



2. 윈도우를 생성하여 합친 이미지를 화면에 출력한다.

3. 생성한 이미지들을 Release하고 윈도우를 Destroy한다.

♋ **실행 결과 (ReportCode2-1.cpp)**



Clamping과 더불어 좌우반전인 두 이미지가 합쳐지는 것을 확인할 수 있다.

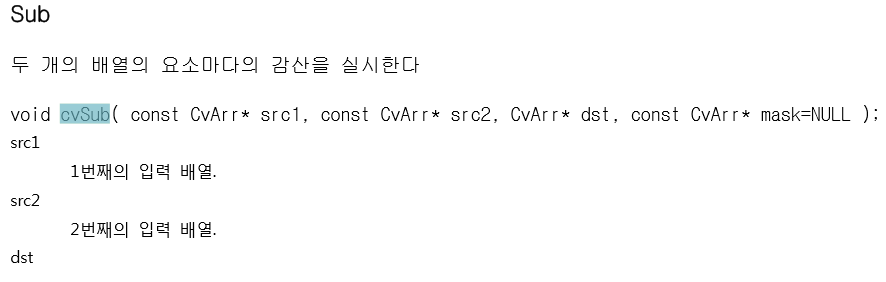
**실습 2)**

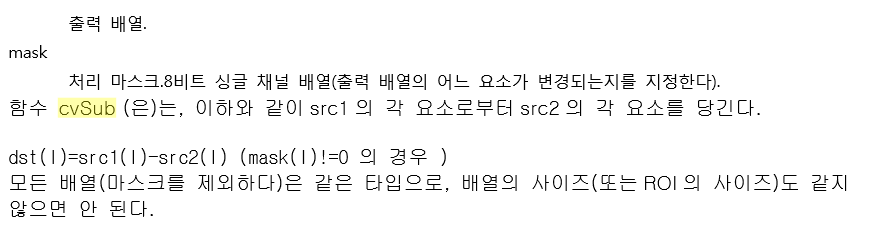
**(2) Subtraction with two images (Input : any two images)**

♋ **동작 순서 및 원리**

0. 사이즈가 똑같은 이미지 2개를 생성한다.

1. 이미지의 두 RGB 값을 빼기 위해 (Clamping 처리 포함) Opencv에서 제공하는 함수 CvSub를 사용한다. 뺄셈 순서는 image.jpg – processed.jpg = 결과 이미지 이다. CvSub의 내용은 다음과 같다.





2. 윈도우를 생성하여 뺄셈한 이미지를 화면에 출력한다.

3. 생성한 이미지들을 Release하고 윈도우를 Destroy한다.

**♋ 실행 결과 (ReportCode2-2.cpp)**



Clamping과 더불어 image.jpg에서 processed.jpg를 뺀 결과 이미지를 확인할 수 있다.

**실습 3)**

**♋ 동작 및 원리**

0. Mat 클래스 객체로 이미지1과 이미지2를 로드 한다.

1. 두 이미지를 섞고자 하는 비율을 입력한다.

2. 이미지의 row값과 column값을 추출한다.

3. row값과 column값을 이용하여 이중 For문을 구현한다.

4. Mat 클래스의 at함수를 이용하여 2개의 이미지의 RGB채널값을 각각 추출한다.

5. 각자 추출한 두 이미지의 RGB채널값을 다음과 같이 비율에 맞게 계산한다.

auto BlendedBlue = (chBlue1 \* ratio + chBlue2 \* (1 - ratio));//image1과 image2의 Blue Channel에서 입력한 비율만큼 Blending

auto BlendedGreen = (chGreen1 \* ratio + chGreen2 \* (1 - ratio));//image1과 image2의 Green Channel에서 입력한 비율만큼 Blending

auto BlendedRed = (chRed1 \* ratio + chRed2 \* (1 - ratio));//image1과 image2의 Red Channel에서 입력한 비율만큼 Blending

6. Clamping을 코드로 구현한다. (값이 255가 넘을 경우 255 대입, 값이 0보다 작을 경우 0 대입)

7. 결과 이미지에 계산된 RGB채널값을 대입한 후 윈도우 창으로 출력한다.

**♋ 실행 결과 (ReportCode2-3.cpp)**

