KANGWON NATIONAL UNIVERSITY

컴퓨터비전 실습

실습2 | Mat

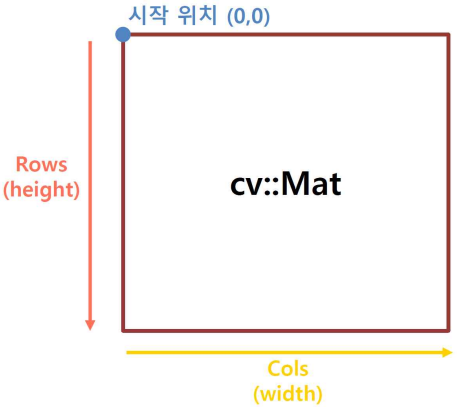
CVMIPLAB @ KNU

강의 진행 및 과제 제출

- 문의방법
  - 컴퓨터비전 TA: 박진오, 이재훈 / 컴퓨터 비전 연구실
  - 이루리 내 Q&A 게시판에 질문 작성
- 과제 제출 방법
  - **결과 이미지 파일과 main 포함된 cpp파일을 이루리 과제 제출란에 압축하여 제출합니다.**

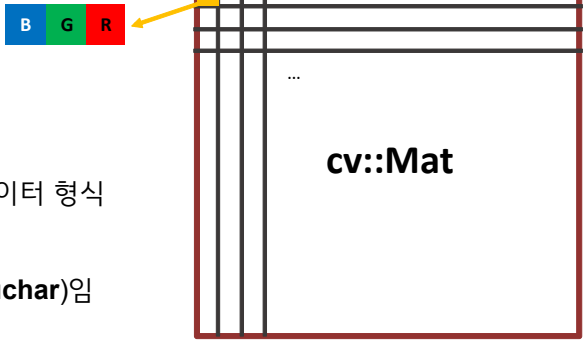
Mat

- OpenCV에서 사용하는 이미지 단위
  - 왼쪽 상단부터 시작하여,
    왼쪽에서 오른쪽 방향으로,
    위에서 아래 방향으로 픽셀 값을 저장함
  - Mat 객체에 접근하여
    이미지를 픽셀 단위나 채널 단위로
    값을 확인하거나 바꿀 수 있음



Mat

- 이미지 저장 단위
  - 각 픽셀마다 B,G,R 의 순서로 값이 저장되어 있음
    (R,G,B가 아님!!)
  - 기본적으로 값이 저장되는 데이터 형식은
    8비트인 **unsigned char** 형(**uchar**)임
    (범위: 0~255)



Mat

- At을 통한 배열의 요소 접근

  Gray scale : image.at<uchar>(y,x)

  Color : Image.at<cv::Vec3b>(y,x)[channel]

- 배열의 각 인덱스로 

각각의 요소에 접근할 수 있다.

  → image.at<Vec3b>(y, x)[0] // 첫번째 컬러 도메인 (B)

  → image.at<Vec3b>(y, x)[1] // 두번째 컬러 도메인 (G)

  → image.at<Vec3b>(y, x)[2] // 세번째 컬러 도메인 (R)

Mat

- 이미지 저장 단위

```
#include "opencv2/opencv.hpp"


int main(void)
{

    // dog.jpg를 읽어 할당합니다.

    cv ::Mat image = cv::imread("dog.jpg", cv::IMREAD_COLOR);

    // 이미지 로드에 실패할 경우를 에러메세지 반환
    if (image.empty()) {
        std::cout << "image load fail" << std::endl;

        return -1;
    }

    // 이미지 최상단 좌측의 픽셀 한 개의 R,G,B값을 읽고 출력한다.
    int x = 0, y = 0;
    std::cout << "Pixel Value:" << image.at<cv::Vec3b>(y, x) << " ";
    std::cout << "B:" << (unsigned int)image.at<cv::Vec3b>(y, x)[0] << " ";
    std::cout << "G:" << (unsigned int)image.at<cv::Vec3b>(y, x)[1] << " ";
    std::cout << "R:" << (unsigned int)image.at<cv::Vec3b>(y, x)[2] << " ";

    return 0;
}
```

```
Pixel Value:[181, 177, 172] B:181 G:177 R:172
```

※ int형으로 casting하지 않고 출력시 char형으로 출력된다

Mat

- 그레이 스케일 표현 (Grayscale)

```cpp
#include "opencv2/opencv.hpp"

int main(void)
{

    // dog.jpg를 읽어 할당합니다.

    cv ::Mat image = cv::imread("dog.jpg", cv::IMREAD_GRAYSCALE);

    // 이미지 로드에 실패할 경우를 에러메세지 반환
    if (image.empty()) {
        std::cout << "image load fail" << std::endl;

        return -1;
    }

    // 이미지 최상단 좌측의 픽셀값을 출력한다.
    int x = 0, y = 0;
    std::cout << "Pixel Value:" << (unsigned int)image.at<uchar>(y, x) << " ";

    return 0;
}
```

```
Pixel Value:176
```

Mat

- 이미지 연산 예제 – 밝기 조절
  - 이미지를 읽어와 밝기를 조절하여 출력한다.
  - **cv::saturate_cast<uchar>(a)**
    - 변수 **a**를 uchar 값의 범위 (0~255 사이)로 clip한다.

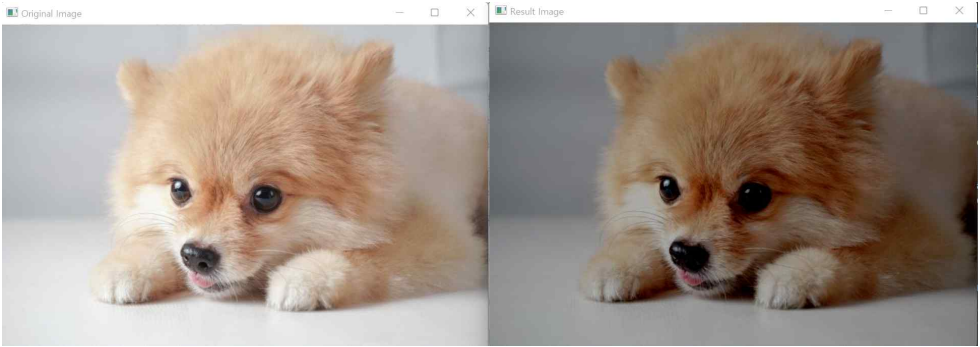
      (예: a=280일때 결과 → a=255,

      a=-16일때 결과 → a=0)

```cpp
#include "opencv2/opencv.hpp"

int main(void)
{

    // 결과를 이미지를 저장할 result선언 및 dog.jpg를 읽어 할당합니다.
    cv::Mat result_image , image;
    image = cv::imread("dog.jpg", cv::IMREAD_COLOR);

    // 이미지를 result에 복사
    image.copyTo(result_image);

    // 이미지 로드에 실패할 경우를 에러메세지 반환
    if (image.empty()) {
        std::cout << "image load fail" << std::endl;

        return -1;
    }

    // 이미지의 전체 밝기 값을 75 낮춘다.
    for (int y = 0; y < image.rows; y++)
    {
        for (int x = 0; x < image.cols; x++)
        {
            // image로부터 픽셀 값을 읽어옴과 동시에 수정한다.
            int b = image.at<cv::Vec3b>(y, x)[0] - 75;
            int g = image.at<cv::Vec3b>(y, x)[1] - 75;
            int r = image.at<cv::Vec3b>(y, x)[2] - 75;

            // 값을 0-255 범위에 맞게 clip하여 result_image에 할당한다
            result_image.at<cv::Vec3b>(y, x)[0] = cv::saturate_cast<uchar>(b);
            result_image.at<cv::Vec3b>(y, x)[1] = cv::saturate_cast<uchar>(g);
            result_image.at<cv::Vec3b>(y, x)[2] = cv::saturate_cast<uchar>(r);

        }
    }

    // 이미지를 서로 비교한다.
    cv::imshow("Original Image", image);
    cv::imshow("Result Image", result_image);
    cv::waitKey(0);

    return 0;
}
```

Mat

- 이미지 연산 예제 – 밝기 조절 결과 화면



실습 2-1 | 이미지 반전

문제

주어진 “.bmp” 파일을 읽고, 이미지를 반전합니다.

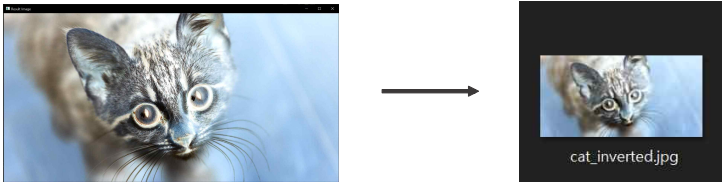
반전된 이미지를 파일로 저장하세요. 저장 파일명은 “cat_inverted.jpg”입니다.

Ex) 255 -> 0 , 0->255

요구 결과

결과 파일 “cat_inverted.jpg”과 main함수가 포함된 “.cpp” 코드 두 개를 압축해 제출합니다.

실행결과



실습 2-2 | 이진화 구현

문제

주어진 ".bmp" 파일을 **흑백으로** 읽고, 이미지를 이진화하여 파일 "cat_binarized.jpg"로 저장합니다.

여기서 이진화란 0부터 255까지로 분포된 값들을 0과 255 두 가지 값만 가지도록 하는 연산으로 정의합니다.

(보통은 0과 1로 변환하는 작업을 의미하나 편의상 0과 255로 하도록 하겠습니다.)

다음 수식에 따라 이진화를 수행하세요: $P^{y,x} = \{P_B^{y,x}, P_G^{y,x}, P_R^{y,x}\}$일 때, $bin(P^{y,x}) = \begin{cases} 0 \ (P^{y,x} < 128) \\ 255 (P^{y,x} \geq 128) \end{cases}$

요구 결과

결과 파일 "cat_binarized.jpg"과 main함수가 포함된 ".cpp" 코드 두 개를 압축해 제출합니다.

실행결과 및 코드

참고: 파일을 흑백으로 읽기 위해서는 IMREAD_GRAYSCALE을 사용합니다.

Q&A

Q1. 결과 이미지가 깨져서 나오는 경우

→ 각 픽셀의 B, G, R 값은 0~255 사이의 범위를 가집니다.
범위가 넘어가면 underflow나 overflow 문제가 발생합니다.
해당 문제를 해결하기 위해 더 큰 자료형에 담아두고 계산한 뒤,
cv::saturate_cast<uchar> 함수를 사용합니다.

Q2. 이미지 파일 저장 방법

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

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