


1

### 영상 분할

□ 영상 분할(image segmentation)이란 영상 안의 화소를 의미 있는 영역(segment)으로 분할하는 것을 의미한다.



Sky

Building

Road

Sidewalk

Fence

Vegetation

Pole

Car

Sign


Pedestrian

Cyclist

출처: Nvidia dev blog

OpenCV로 배우는 디지털 영상 처리

© 인피니티박스 2019



2

## 영상 분할 방법

- 에지(edge)를 사용하여 물체의 윤곽선을 추적
- 클러스터링을 이용하여 영역 성장법(region growing) 사용
- 히스토그램이나 무늬를 이용하여 분할하는 방법
- 딥러닝(deep learning)을 이용해서 영상을 분할하는 방법

OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019



3

## 이진화

- 어떤 임계값을 정하고 이 값을 기준으로 그레이스케일 영상을 이진 영상으로 만든다.
- 최대 엔트로피 방법(maximum entropy method)
- 오투의 방법(Otsu's method)
- k-means 클러스터링

OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019



4

## 이진화



CC BY 1.0, <https://commons.wikimedia.org/w/index.php?curid=10634721>

OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019



5

## 클러스터링 방법(Clustering methods)

1. 임의로 K개 클러스터의 중심을 선택한다.
2. 각 화소를 화소와 클러스터 중심 사이의 거리가 최소가 되는 클러스터에 할당한다.
3. 클러스터의 모든 화소를 평균하여 클러스터 중심을 다시 계산한다.
4. 클러스터링이 수렴할 때까지 2단계와 3단계를 반복한다.

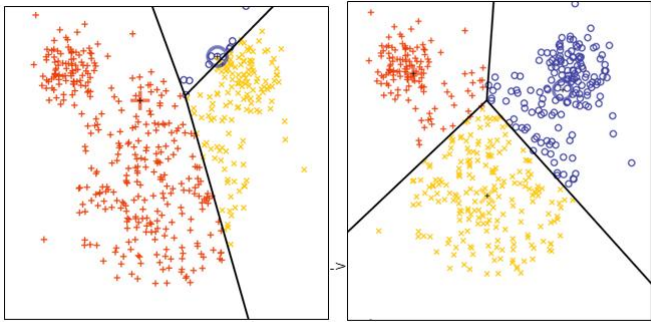
OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019



6

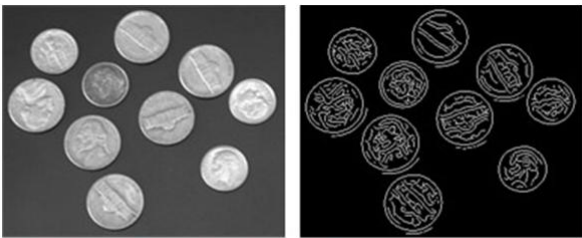
# 클러스터링 방법(Clustering methods)



(출처: 워키 백과)



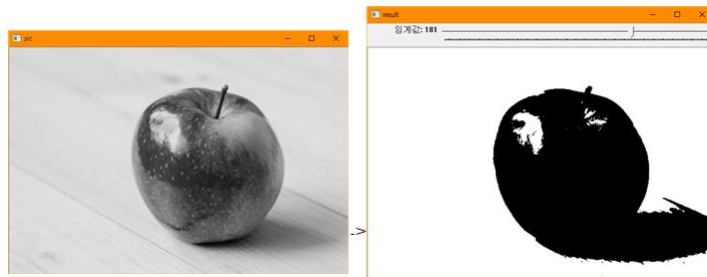
# 에지 기반 방법



영상 출처: <https://kr.mathworks.com/discovery/edge-detection.html>



## 이진화



OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019



9

## 이진화

$$dst(x, y) \simeq \begin{cases} 1, & src(x, y) > T \text{인 경우} \\ 0, & src(x, y) \leq T \text{인 경우} \end{cases}$$

OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019



10

## 이진화

threshold(src, dst, thresh, maxval, type)

매개 변수	설명
src	입력 영상
dst	출력 영상
thresh	임계값
maxval	화소값이 임계값을 넘으면 부여되는 값
type	이진화 타입. 5개 중에서 하나이다. <ul style="list-style-type: none"> <li>• THRESH_BINARY</li> <li>• THRESH_BINARY_INV</li> <li>• THRESH_TRUNC</li> <li>• THRESH_TOZERO</li> <li>• THRESH_TOZERO_INV</li> </ul>

OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019



11

```

int threshold_value = 128;
int threshold_type = 0;
const int max_value = 255;
const int max_binary_value = 255;
Mat src, src_gray, dst;

static void MyThreshold(int, void*)
{
    threshold(src, dst, threshold_value, max_binary_value, threshold_type);
    imshow("result", dst);
}

int main()
{
    src = imread("d:/lenna.jpg", IMREAD_GRAYSCALE);
    namedWindow("result", WINDOW_AUTOSIZE);
    createTrackbar("임계값",
        "result", &threshold_value,
        max_value, MyThreshold);
    MyThreshold(0, 0); // 초기화를 위하여 호출한다.
    waitKey();
    return 0;
}

```

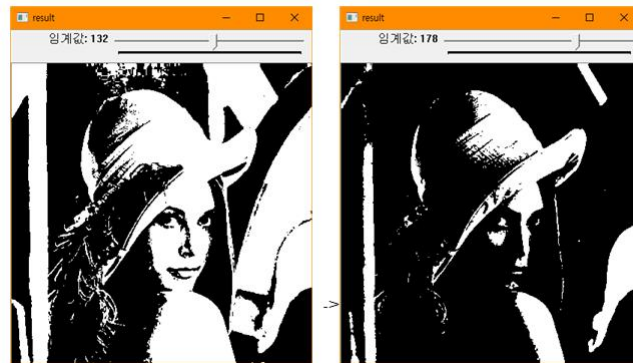
OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019



12

## 실행 결과



OpenCV로 배우는 디지털 영상 처리

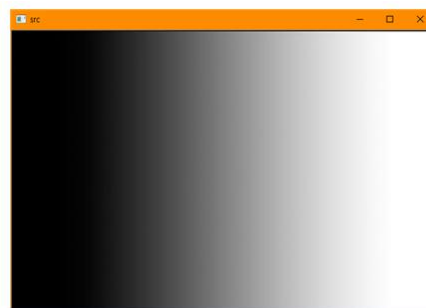
© 인피니티북스 2019



13

## 이진화 타입

- THRESH\_BINARY
- THRESH\_BINARY\_INV
- THRESH\_TRUNC
- THRESH\_TOZERO
- THRESH\_TOZERO\_INV



OpenCV로 배우는 디지털 영상 처리

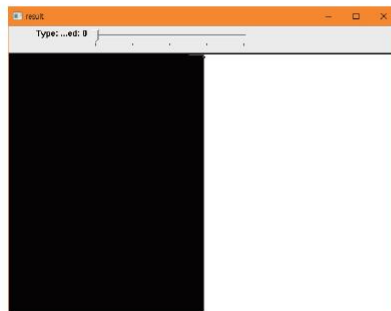
© 인피니티북스 2019



14

## THRESH\_BINARY

$$dst(x, y) = \begin{cases} \text{maxVal} & \text{if } src(x, y) > \text{thresh} \\ 0 & \text{otherwise} \end{cases}$$



OpenCV로 배우는 디지털 영상 처리

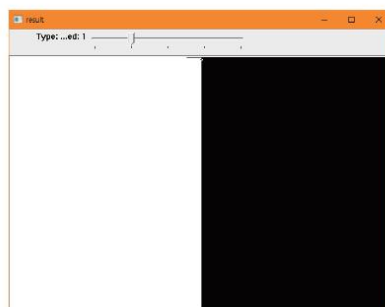
© 인피니티박스 2019



15

## THRESH\_BINARY\_INV

$$dst(x, y) = \begin{cases} 0 & \text{if } src(x, y) > \text{thresh} \\ \text{maxVal} & \text{otherwise} \end{cases}$$



OpenCV로 배우는 디지털 영상 처리

© 인피니티박스 2019

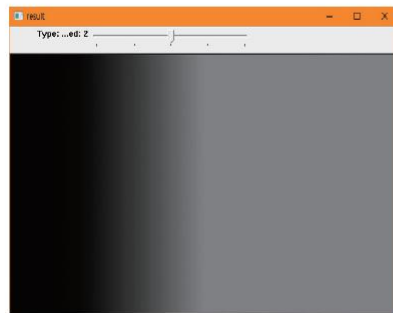


16



## THRESH\_TRUNC

$$dst(x, y) = \begin{cases} thresh & \text{if } src(x, y) > thresh \\ src(x, y) & \text{otherwise} \end{cases}$$



OpenCV로 배우는 디지털 영상 처리

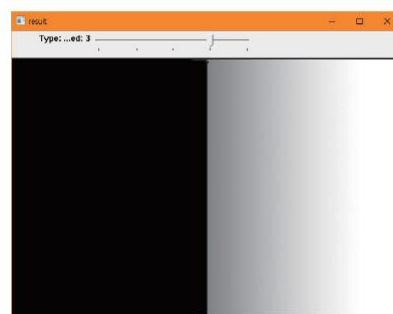
© 인피니티박스 2019



17

## THRESH\_TOZERO

$$dst(x, y) = \begin{cases} src(x, y) & \text{if } src(x, y) > thresh \\ 0 & \text{otherwise} \end{cases}$$



OpenCV로 배우는 디지털 영상 처리

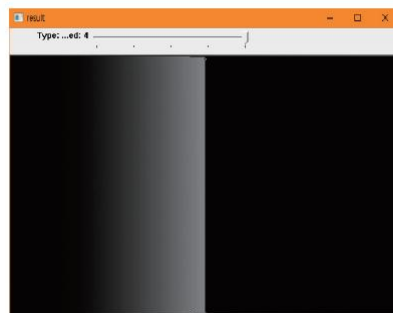
© 인피니티박스 2019



18

## THRESH\_TOZERO\_INV

$$dst(x, y) = \begin{cases} 0 & \text{if } src(x, y) > \text{thresh} \\ src(x, y) & \text{otherwise} \end{cases}$$



OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019



19

## 이진화

```
#include "opencv2/opencv.hpp"
#include <iostream>
using namespace std;
using namespace cv;

int threshold_value = 128;
int threshold_type = 0;
const int max_type = 4;
const int max_binary_value = 255;
Mat src, src_gray, dst;

static void MyThreshold(int, void*)
{
    threshold(src, dst, threshold_value, max_binary_value, threshold_type);
    imshow("result", dst);
}
```

OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019



20

## 이진화

```
int main()
{
    src = imread("d:/lenna.jpg", IMREAD_GRAYSCALE);
    imshow("src", src);
    namedWindow("result", WINDOW_AUTOSIZE);
    createTrackbar("Type: \n 0: Binary \n 1: Binary Inverted \n 2: Truncate \n 3: To Zero \n 4: To Zero Inverted",
        "result", &threshold_type,
        max_type, MyThreshold);

    MyThreshold(0, 0); // 초기화를 위하여 호출한다.
    waitKey();
    return 0;
}
```

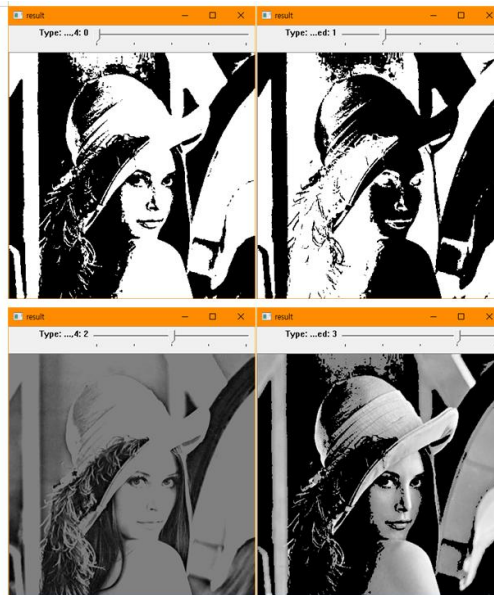
OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019



21

## 실행 결과



OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019



22

적응적 이진화

Original Image

Global Thresholding (v = 127)

Adaptive Mean Thresholding

Adaptive Gaussian Thresholding

OpenCV로 배우는 ... 출처: OpenCV 튜토리얼

23

적응적 이진화

OpenCV로 배우는 디지털 영상 처리 © 인피니티북스 2019

24


적응적 이진화

□ Chow와 Kaneko 접근 방식

□ 지역 임계값 방식

OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019




적응적 이진화

```
void adaptiveThreshold(InputArray src, OutputArray dst, double maxValue,
                      int adaptiveMethod)
```

매개 변수	설명
src	입력 영상
dst	출력 영상
maxValue	화소값이 임계값을 넘으면 부여되는 값
adaptiveMethod	적응적 이진화의 방법 선택
ADAPTIVE_THRESH_MEAN_C	임계값은 인접 지역의 평균이 된다.
ADAPTIVE_THRESH_GAUSSIAL_C	임계값은 가중치가 가우시안인 윈도우를 이웃 화소에 씌워서 계산한 가중치의 합이다.

OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019



## 이진화

```
int main()
{
    Mat src = imread("d:/book1.jpg", IMREAD_GRAYSCALE);
    Mat img, th1, th2, th3, th4;
    medianBlur(src, img, 5);
    threshold(img, th1, 127, 255, THRESH_BINARY);
    adaptiveThreshold(img, th2, 255, ADAPTIVE_THRESH_MEAN_C,
    THRESH_BINARY, 11, 2);
    adaptiveThreshold(img, th3, 255, ADAPTIVE_THRESH_GAUSSIAN_C,
    THRESH_BINARY, 11, 2);

    imshow("Original", src);
    imshow("Global Thresholding", th1);
    imshow("Adaptive Mean", th2);
    imshow("Adaptive Gaussian", th3);
    waitKey();
    return 0;
}
```

OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019



27

## 실행 결과



전역이진화

Adaptive Mean 방법

Adaptive Gaussian 방법

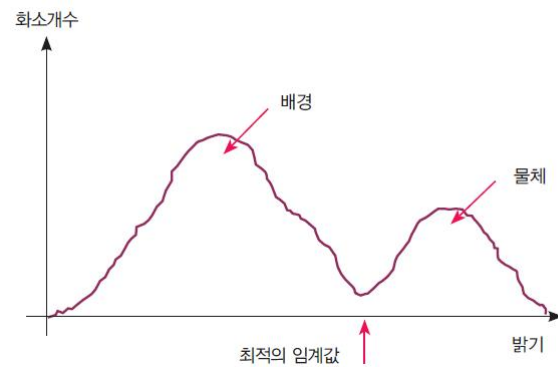
OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019



28

## Otsu의 이진화 방법



OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019



29

## Otsu의 이진화 방법

- Otsu의 방법에서는 두 클래스의 가중치 합계로 정의되는 클래스 내 분산을 최소화하는 임계값을 검색한다.

$$\sigma_w^2 = W_b \sigma_b^2 + W_f \sigma_f^2$$

OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019



30

이 지 하

```
int main()
{
    Mat src = imread("d:/lenna.jpg", IMREAD_GRAYSCALE); // Load an image
    Mat blur, th1, th2, th3, th4;
    threshold(src, th1, 127, 255, THRESH_BINARY);
    threshold(src, th2, 0, 255, THRESH_BINARY | THRESH_OTSU);

    Size size = Size(5, 5);
    GaussianBlur(src, blur, size, 0);
    threshold(blur, th3, 0, 255, THRESH_BINARY | THRESH_OTSU);

    imshow("Original", src);
    imshow("Global", th1);
    imshow("Ostu", th2);
    imshow("Ostu after Blurring", th3);
    waitKey();
    return 0;
}
```

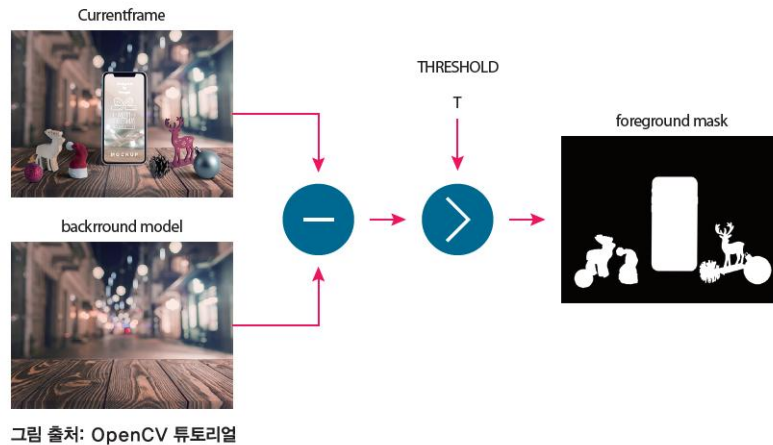


실 행 결 과





## 배경 제거



OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019



33

## BackgroundSubtractorMOG2

- 이것은 가우시안 혼합 기반 배경 / 전경 분할 알고리즘이다.
- 이 방법은 2004년도의 Z. Zivkovic, 「*Improved adaptive Gaussian mixture model for background subtraction*」 논문과 2006년도의 「*Efficient Adaptive Density Estimation per Image Pixel for the Task of Background Subtraction*」 논문에서 소개되었다

OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019



34

```

int main()
{
    Mat frame; // 현재 프레임
    Mat result; // MOG2에 의하여 생성되는 결과 영상
    Ptr<BackgroundSubtractor> pMOG2; //MOG2 배경 삭제 객체
    int keyboard;

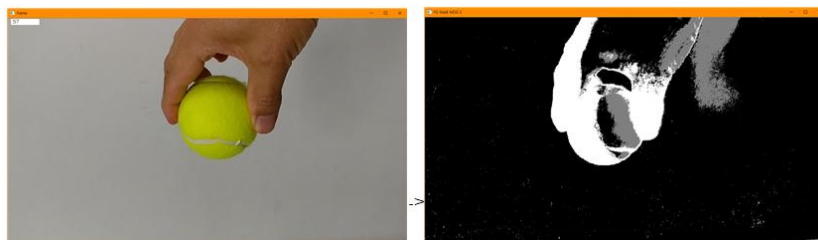
    pMOG2 = createBackgroundSubtractorMOG2();
    VideoCapture capture("d:/tennis_ball.mp4");
    if (!capture.isOpened()) { exit(EXIT_FAILURE); }

    while ((char)keyboard != 27) {
        if (!capture.read(frame)) {
            exit(EXIT_FAILURE);
        }
        pMOG2->apply(frame, result);
        imshow("Frame", frame);
        imshow("FG Mask MOG 2", result);
        keyboard = waitKey(30);
    }
    capture.release();
    return 0;
}

```

35

## 실행 결과



OpenCV로 배우는 디지털 영상 처리

© 인피니티박스 2019



36

연결 성분 레이블링

			1	1	1								3	3	3	3			
		1	1	1														3	
								2	2	2	2	2						3	
			2	2	2	2	2	2	2	2	2	2						3	
			2										2					3	
			2	2									2					3	
				2									2						
				2	2	2	2	2	2	2	2	2							

배경

전경

OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019

4-연결과 8-연결

	•	
•	•	

•	•	•
•	•	

OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019

## 제귀 알고리즘

- ① 영상을 스캔하여 레이블링되어 있지 않은 전경 화소를 찾아 새로운 레이블  $L$ 을 부여한다.
- ② 재귀적으로 레이블  $L$ 을 모든 이웃의 전경 화소(4-이웃 또는 8-이웃)에 부여한다.
- ③ 더 이상 레이블링되어 있지 않은 전경 화소가 없으면 멈춘다.
- ④ 단계 ①로 간다.



## 2-패스 알고리즘

### Algorithm 11.2

알고리즘: 4-연결을 이용한 2-패스 연결 성분 레이블링

첫 번째 패스

- ① 영상을 위에서 아래로, 왼쪽으로 오른쪽으로 스캔한다.
- ② 현재 화소가 1이면
  - (a) 왼쪽 화소만이 레이블을 가지면 그 레이블을 현재 화소에 부여한다.
  - (b) 위쪽 화소만이 레이블을 가지면 그 레이블을 현재 화소에 부여한다.
  - (c) 위쪽과 왼쪽 화소가 다른 레이블을 가지면 이 사실을 등가 테이블에 기록한다.
  - (d) 위의 경우가 아니면 이 화소에 새로운 레이블을 부여한다.
- ③ 고려해야 할 더 이상의 화소가 없으면 멈춘다.

두 번째 패스

- ① 등가 테이블에서 각 등가 레이블 집합에서 최소의 레이블을 찾는다.
- ② 영상을 조사하여 레이블을 등가 집합의 최소 레이블로 바꾼다.

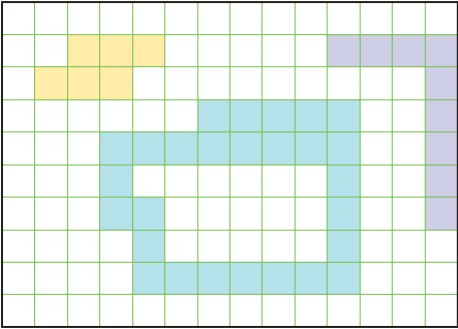
4-connected

	2	
1	•	





## 2-패스 알고리즘




배경

전경

OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019



43

## 2-패스 알고리즘

```
int connectedComponentsWithStats(InputArray image, OutputArray labels,
                                OutputArray stats, OutputArray centroids, int connectivity=8,
                                int ltype=CV_32S)
```

매개 함수	설명
image	입력 영상
labels	레이블 영상
connectivity	8-연결성이나 4-연결성
ltype	출력 영상의 레이블 타입 CV_32S 또는 CV_16U
statsv	각 레이블에 대한 통계 자료
반환값	레이블의 개수

OpenCV로 배우는 디지털 영상 처리

© 인피니티북스 2019



44

## 이진화

```
int main() {
    Mat img, img_edge, labels, centroids, img_color, stats;
    img = cv::imread("d:/coins.png", IMREAD_GRAYSCALE);

    threshold(img, img_edge, 128, 255, THRESH_BINARY_INV);
    imshow("Image after threshold", img_edge);

    int n = connectedComponentsWithStats(img_edge, labels, stats, centroids);

    vector<Vec3b> colors(n + 1);
    colors[0] = Vec3b(0, 0, 0);
    for (int i = 1; i <= n; i++) {
        colors[i] = Vec3b(rand() % 256, rand() % 256, rand() % 256);
    }
}
```

45

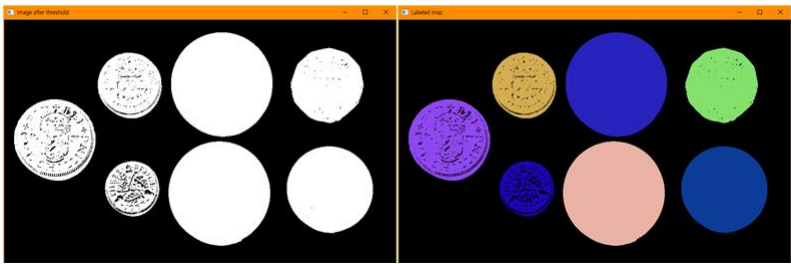
## 이진화

```
img_color = cv::Mat::zeros(img.size(), CV_8UC3);
for (int y = 0; y < img_color.rows; y++)
    for (int x = 0; x < img_color.cols; x++)
    {
        int label = labels.at<int>(y, x);
        img_color.at<cv::Vec3b>(y, x) = colors[label];
    }

cv::imshow("Labeled map", img_color);
cv::waitKey();
return 0;
}
```

46

# 실행 결과

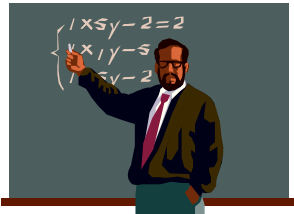


OpenCV로 배우는 디지털 영상 처리      © 인피니티북스 2019



47

# Q & A



OpenCV로 배우는 디지털 영상 처리      © 인피니티북스 2019



48