OPEN CV 활용

(주요 명령어 사용법 , 영상파일 입출력 및 변환 등)

2022. 03. 23

고 정 재 / 유 용 주

1.Open CV란?

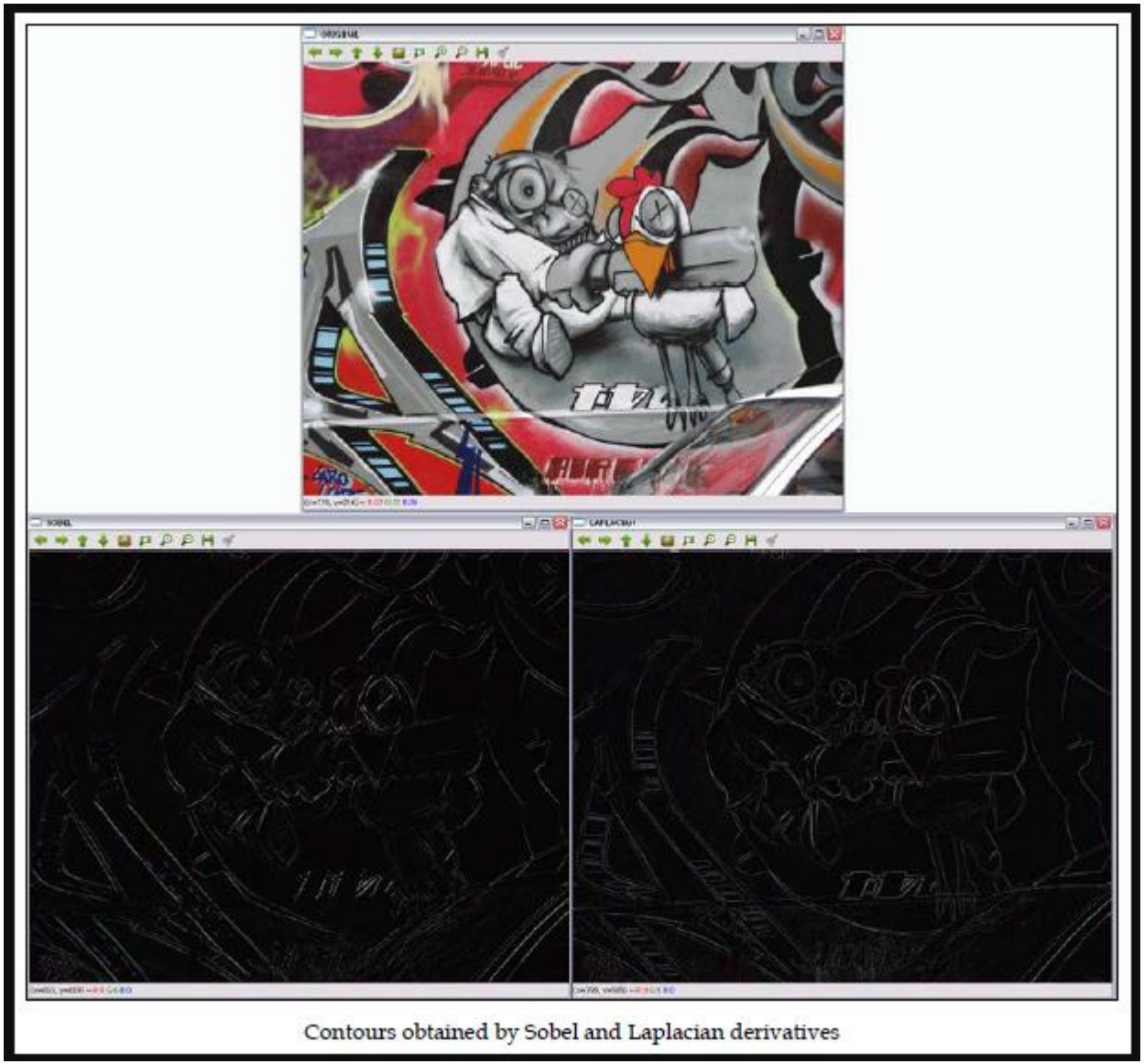
Open Source Computer Vision의 약자로 다양한 영상/동영상 처리에 사용할 수 있는 오픈소스 라이브러리 입니다.

OpenCV는 BSD 라이센스 하에 배포되므로 학술적 및 상업적 용도로 무료입니다. 따라서 OpenCV를 이용하여 제품을 만들어서 수익이 나도 소스코드를 공유, 라이센스 비용 지불을 하지 않아 많은 기업이나 개인 개발자가 사용하기도 합니다.
"BSD (Berkeley Software Distribution) 라이선스는 소프트웨어 라이선스라고도 할 수 없을 만큼 미약하여, 해당 소프트웨어는 아무나 개작할 수 있고, 수정한 것을 제한 없이 배포할 수 있다. 다만 수정본의 재배포는 의무적인 사항이 아니므로 BSD 라이선스를 갖는 프로그램은 공개하지 않아도 되는 상용 소프트웨어에서도 사용할 수 있다."
또한 C++, C, Python 및 Java와 같은 다양한 인터페이스를 지원하며 Windows, Linux, Mac OS, iOS 및 Android같은 다양한 OS를 지원합니다.
OpenCV는 알고리즘 상으로 계산 효율성과 실시간 응용 프로그램에 중점을 두고 설계되었기 때문에 간단하게 OpenCV에서 제공되는 API를 사용하여 코딩하여도 실시간 프로세싱이 가능한 어플리케이션을 만들 수 있기 때문에 최적화나 알고리즘을 생각하지 않고도 품질 좋은 상용 프로그램을 만들 수 있습니다.
또한 OpenCV는 멀티 코어 프로세싱을 지원하기 때문에 다양한 상황에 응용이 가능합니다.

interactive art나 image stitching, 공장의 불량품 검출 시스템 및 로봇공학 등 다양한 영상처리 시스템에서 이용되고 있습니다.

2.무엇을 할수 있나?

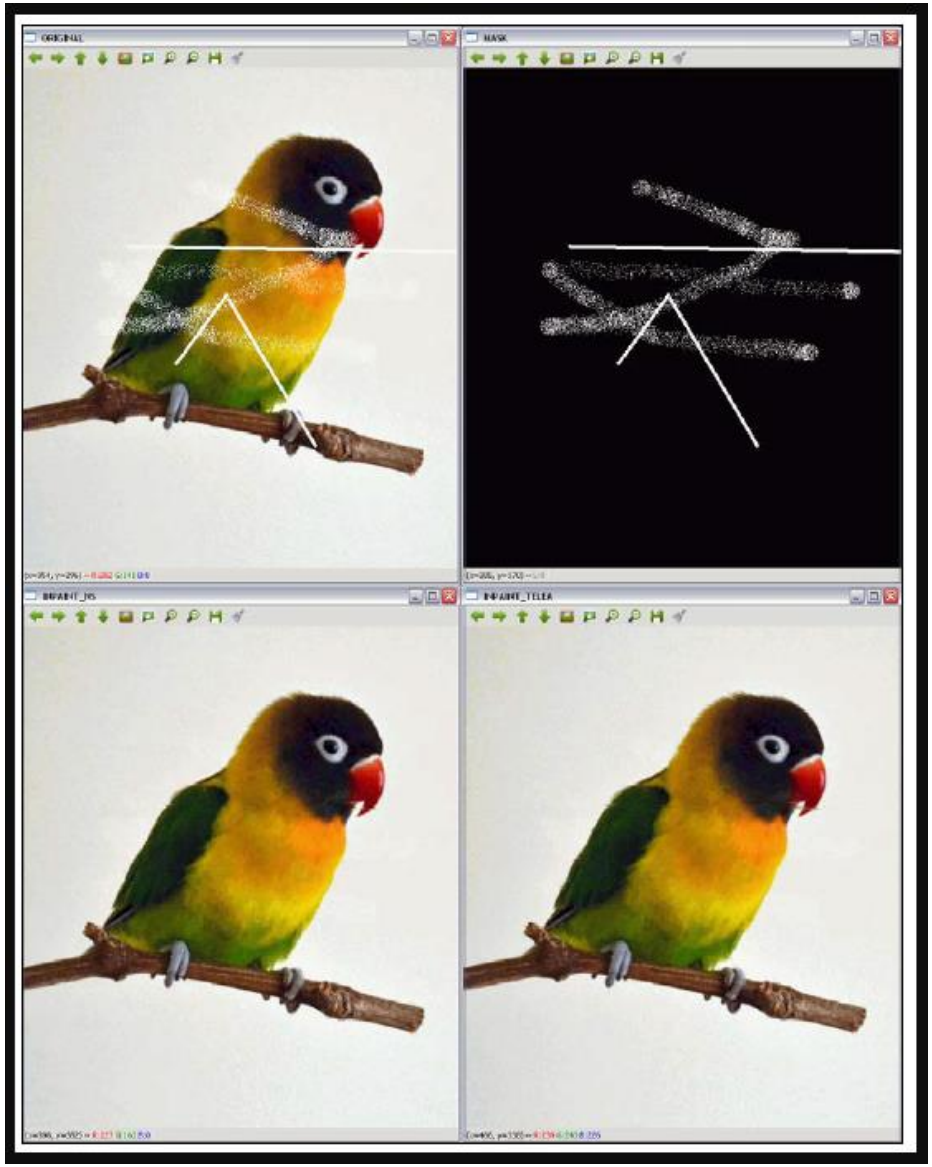
1) 윤곽선 검출



Contours obtained by Sobel and Laplacian derivatives

2.무엇을 할수 있나?

2) 노이즈 제거



2.무엇을 할수 있나?

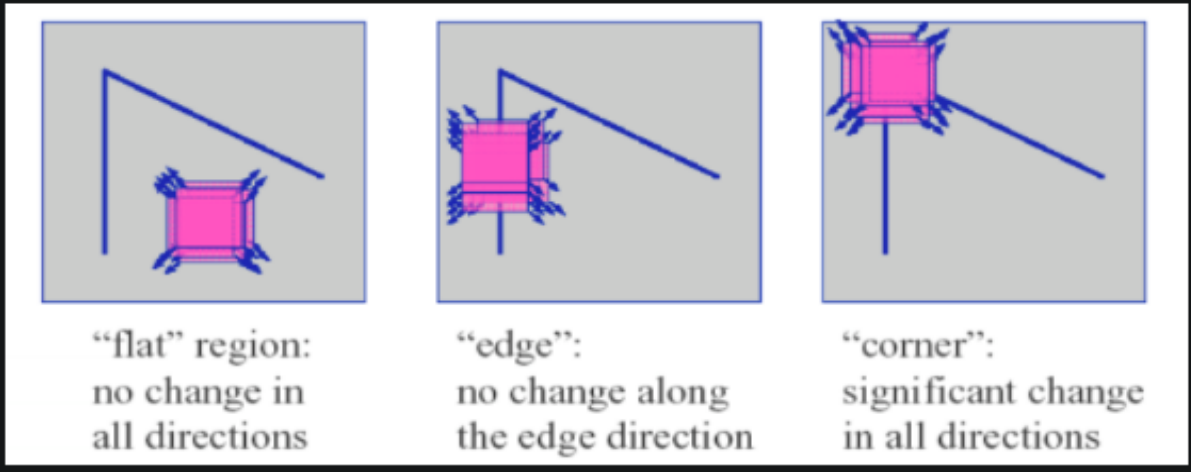
3) 이미지 스티칭을 이용한 파노라믹 사진 제작

2.무엇을 할수 있나?

4) Harris corner detector

영상에서 코너점, 특징점(keypoint)을 찾는 가장 대표적인 방법은 1988년에 발표된 Harris corner detector 입니다.

디지털 이미지는 점들, 즉 픽셀들의 집합으로 구성되어 있습니다. 그 점들 중에서 가장 중요한 점들은 단연 코너점들입니다. 코너점(corner point)이란 두 방향 이상에서 변화가 급격한 점입니다. 반면 엣지점(edge point)은 한 방향에서 변화가 급격한 점입니다. 이도 저도 아닌 점들은 평탄한 점(flat point)이라고 부릅니다.



출처: https://subscription.packtpub.com/book/application_development/9781788299763/3/03lvl1sec17/harris-corner-detection

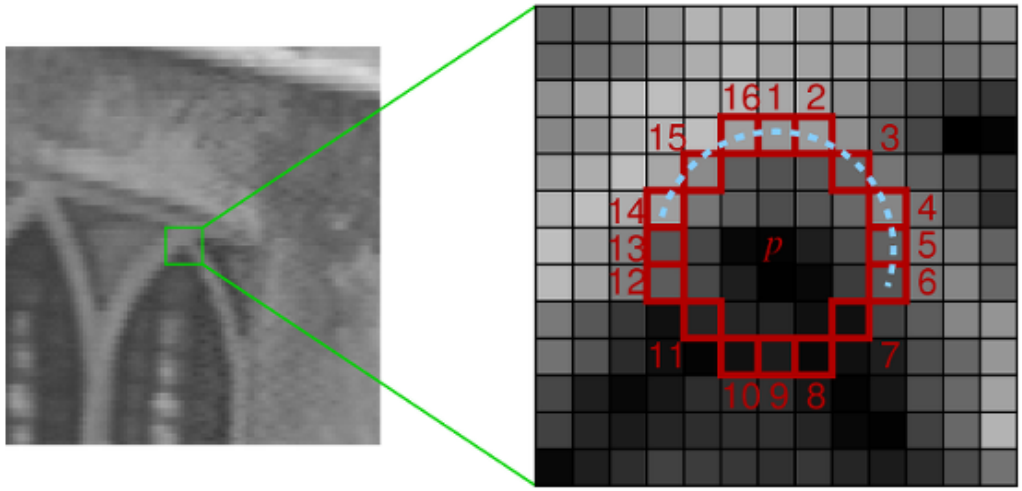
코너점이 중요한 이유는 이미지에서 가장 중요한 구조 정보를 담고 있기 때문입니다. 다른 정보를 모두 없애고 코너점들만 남겨놔도 이미지 내의 물체들의 형상을 대충 알 수 있습니다.

2.무엇을 할수 있나?

5) FAST corner detector

FAST(Features from Accelerated Segment Test)는 영국 캠브리지 대학의 Edward Rosten이 개발한 방법으로 FAST라는 이름에서 알 수 있듯이 극도의 빠름을 추구한 특징점 추출 방법입니다. 하지만 FAST가 정말 뛰어난 점은 FAST가 속도에 최적화되어 설계된 기술임에도 불구하고 그 특징점의 품질(repeatability: 다양한 영상 변화에서도 동일한 특징점이 반복되어 검출되는 정도) 또한 기존의 방법들(Harris, DoG, ...)을 상회한다는 점에 있습니다.

FAST 또한 코너점을 찾는 방법중 하나로서 그 기본 방법은 다음과 같습니다.

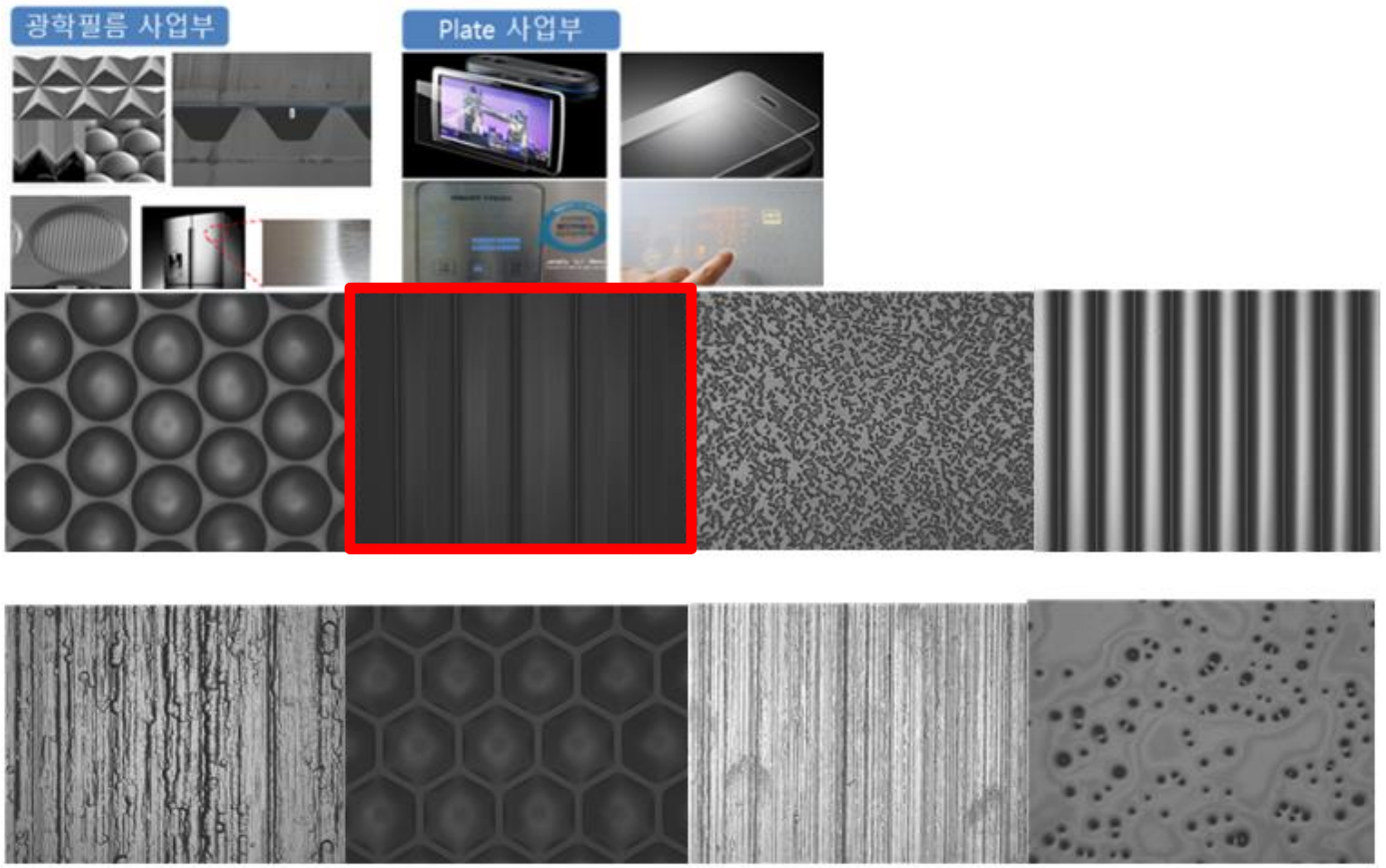


<그림 7> FAST corner detector

FAST에서는 위 그림과 같이 어떤 점 p가 코너(corner)인지 여부를 p를 중심으로 하는 반지름 3인 원 상의 16개 픽셀값을 보고 판단합니다. 그래서 p보다 일정값 이상 밝은 (>p+t) 픽셀들이 n개 이상 연속되어 있거나 또는 일정값 이상 어두운(<p-t) 픽셀들이 n개 이상 연속되어 있으면 p를 코너점으로 판단합니다.

3.당사 제품 패턴이미지로 영상 처리 예시

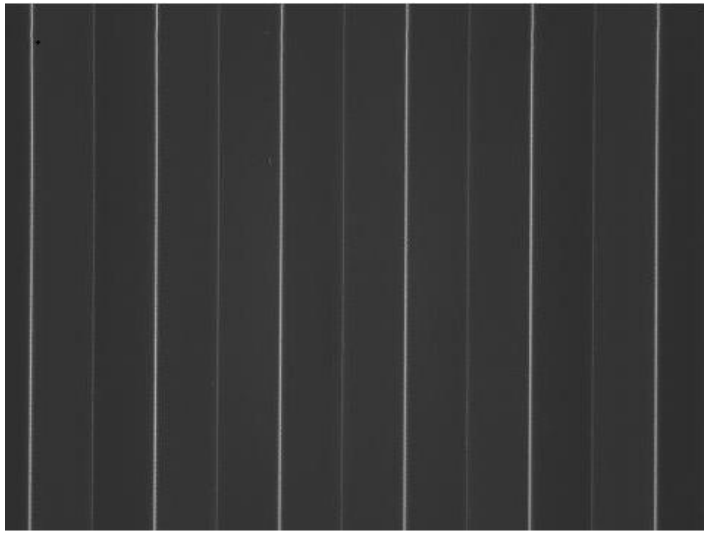
- 당사는 광학필름 및 가전 데코레이션 필름을 개발,제조하는 회사입니다.



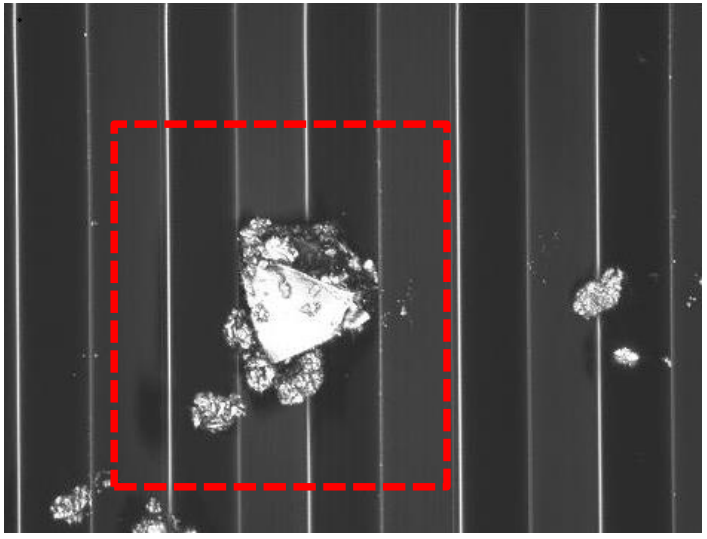
- 위와 같이 다양한 패턴을 가지고 제품을 개발 , 제조하고 있습니다.
- 이 중 광학필름에 한 종류인 Prism패턴을 영상 처리를 했습니다.

4.추출 영상

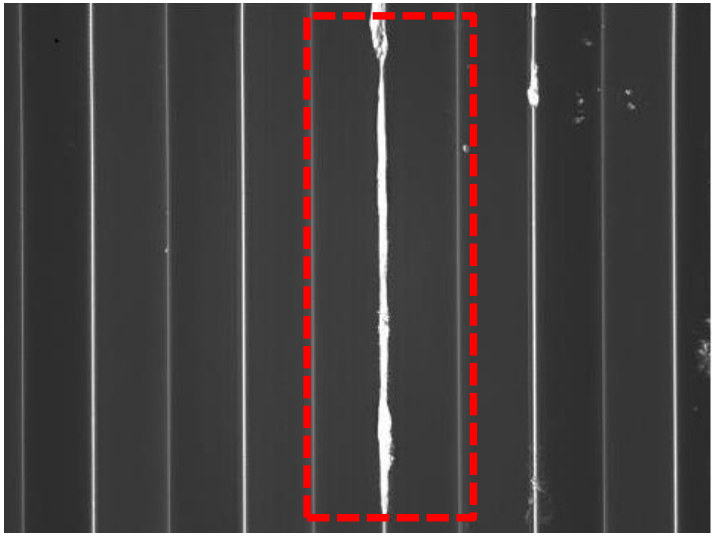
➢ OK 제품



➢ NG 제품



(White spot)



(Scratch)

추출 알고리즘 – Harris corner detector/FAST corner detector

1. OK 제품- code

```
import cv2
import numpy as np

img = cv2.imread('c:/img/OK2.PNG', cv2.IMREAD_COLOR)
corners = cv2.cornerHarris(cv2.cvtColor(img, cv2.COLOR_BGR2GRAY), 2, 3, 0.04)

corners = cv2.dilate(corners, None)

show_img = np.copy(img)
show_img[corners>0.1*corners.max()]=[0,0,255]

corners = cv2.normalize(corners, None, 0, 255, cv2.NORM_MINMAX).astype(np.uint8)
show_img = np.hstack((show_img, cv2.cvtColor(corners, cv2.COLOR_GRAY2BGR)))


cv2.imshow('Harris corner detector', show_img)
if cv2.waitKey(0) == 27:
    cv2.destroyAllWindows()

fast = cv2.FastFeatureDetector_create(30, True, cv2.FAST_FEATURE_DETECTOR_TYPE_9_16)
kp = fast.detect(img)

show_img = np.copy(img)
for p in cv2.KeyPoint_convert(kp):
    cv2.circle(show_img, tuple(p), 2, (0, 255, 0), cv2.FILLED)

cv2.imshow('FAST corner detector', show_img)
if cv2.waitKey(0) == 27:
    cv2.destroyAllWindows()

fast.setNonmaxSuppression(False)
kp = fast.detect(img)

for p in cv2.Keypoint_convert(kp):
    cv2.circle(show_img, tuple(p), 2, (0, 255, 0), cv2.FILLED)

cv2.imshow('FAST corner detector', show_img)
if cv2.waitKey(0) == 27:
   cv2.destroyAllWindows()
```

추출 알고리즘 – Harris corner detector/FAST corner detector

2. White spot NG제품 – code

```
import cv2
import numpy as np

img = cv2.imread('c:/img/white spot.PNG', cv2.IMREAD_COLOR)
corners = cv2.cornerHarris(cv2.cvtColor(img, cv2.COLOR_BGR2GRAY), 2, 3, 0.04)

corners = cv2.dilate(corners, None)

show_img = np.copy(img)
show_img[corners>0.1*corners.max()]=[0,0,255]

corners = cv2.normalize(corners, None, 0, 255, cv2.NORM_MINMAX).astype(np.uint8)
show_img = np.hstack((show_img, cv2.cvtColor(corners, cv2.COLOR_GRAY2BGR)))


cv2.imshow('Harris corner detector', show_img)
if cv2.waitKey(0) == 27:
    cv2.destroyAllWindows()

fast = cv2.FastFeatureDetector_create(30, True, cv2.FAST_FEATURE_DETECTOR_TYPE_9_16)
kp = fast.detect(img)

show_img = np.copy(img)
for p in cv2.KeyPoint_convert(kp):
    cv2.circle(show_img, tuple(p), 2, (0, 255, 0), cv2.FILLED)

cv2.imshow('FAST corner detector', show_img)
if cv2.waitKey(0) == 27:
    cv2.destroyAllWindows()

fast.setNonmaxSuppression(False)
kp = fast.detect(img)

for p in cv2.Keypoint_convert(kp):
    cv2.circle(show_img, tuple(p), 2, (0, 255, 0), cv2.FILLED)

cv2.imshow('FAST corner detector', show_img)
if cv2.waitKey(0) == 27:
   cv2.destroyAllWindows()
```

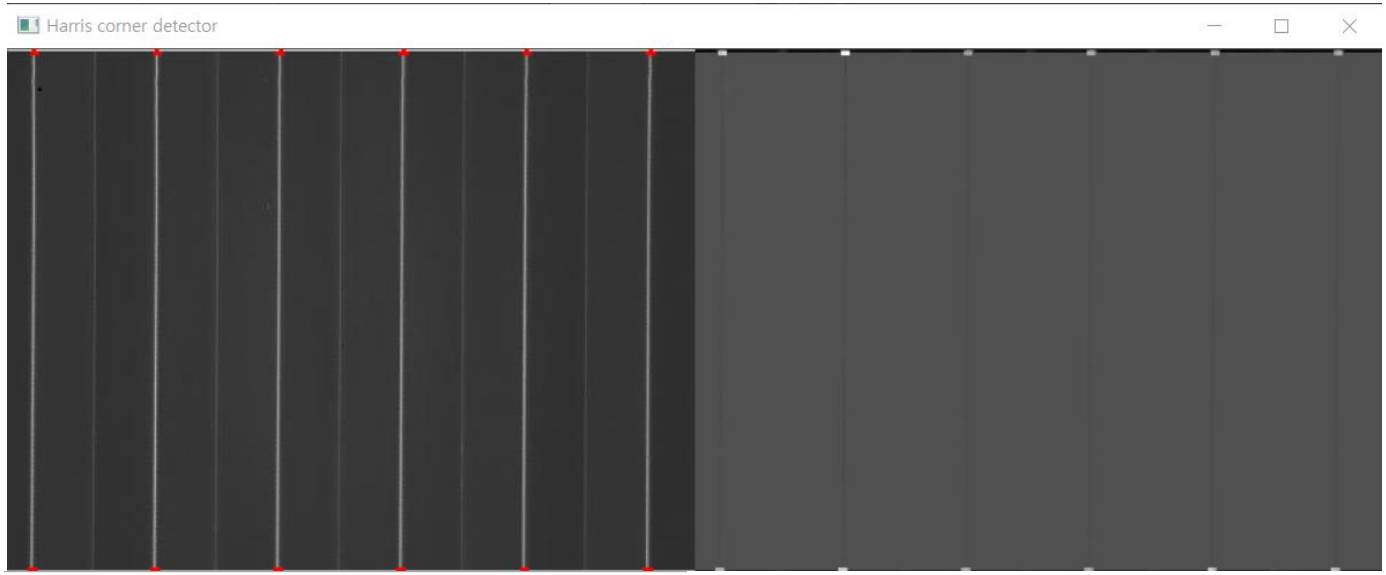
추출 알고리즘 – Harris corner detector/FAST corner detector

3. Scratch NG 제품 – code

```
import cv2
import numpy as np

img = cv2.imread('c:/img/Scratch.PNG', cv2.IMREAD_COLOR)
corners = cv2.cornerHarris(cv2.cvtColor(img, cv2.COLOR_BGR2GRAY), 2, 3, 0.04)

corners = cv2.dilate(corners, None)

show_img = np.copy(img)
show_img[corners>0.1*corners.max()]=[0,0,255]

corners = cv2.normalize(corners, None, 0, 255, cv2.NORM_MINMAX).astype(np.uint8)
show_img = np.hstack((show_img, cv2.cvtColor(corners, cv2.COLOR_GRAY2BGR)))


cv2.imshow('Harris corner detector', show_img)
if cv2.waitKey(0) == 27:
    cv2.destroyAllWindows()

fast = cv2.FastFeatureDetector_create(30, True, cv2.FAST_FEATURE_DETECTOR_TYPE_9_16)
kp = fast.detect(img)

show_img = np.copy(img)
for p in cv2.KeyPoint_convert(kp):
    cv2.circle(show_img, tuple(p), 2, (0, 255, 0), cv2.FILLED)

cv2.imshow('FAST corner detector', show_img)
if cv2.waitKey(0) == 27:
    cv2.destroyAllWindows()

fast.setNonmaxSuppression(False)
kp = fast.detect(img)

for p in cv2.Keypoint_convert(kp):
    cv2.circle(show_img, tuple(p), 2, (0, 255, 0), cv2.FILLED)

cv2.imshow('FAST corner detector', show_img)
if cv2.waitKey(0) == 27:
   cv2.destroyAllWindows()
```

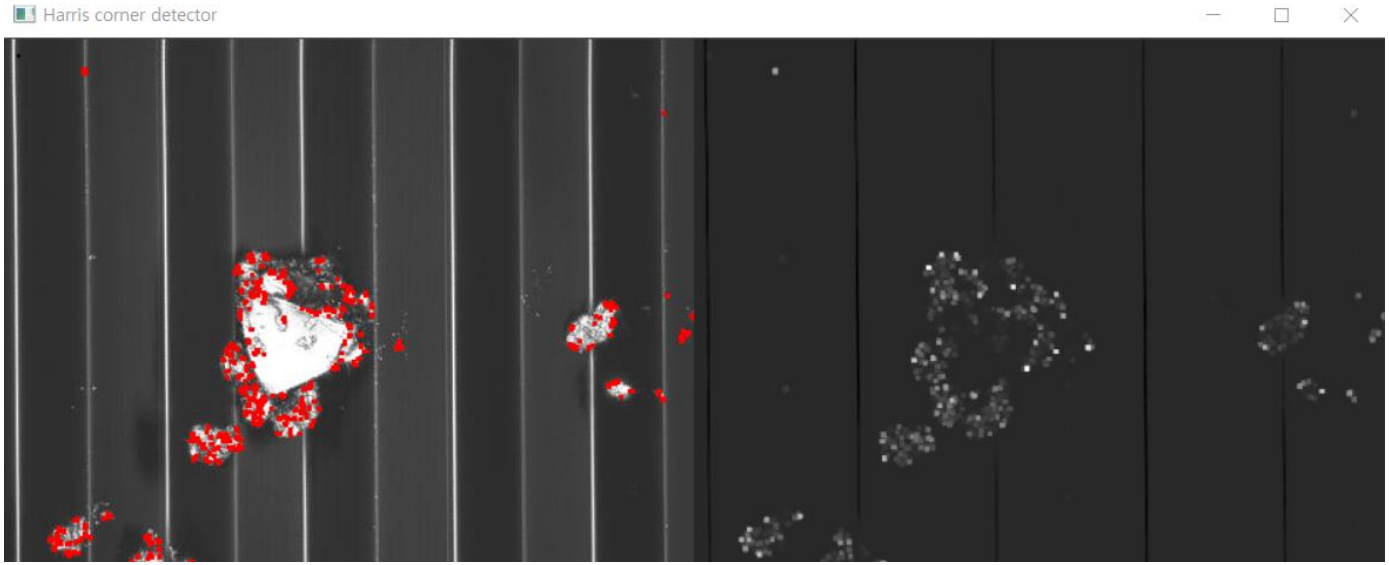
추출 알고리즘 – Harris corner detector/FAST corner detector

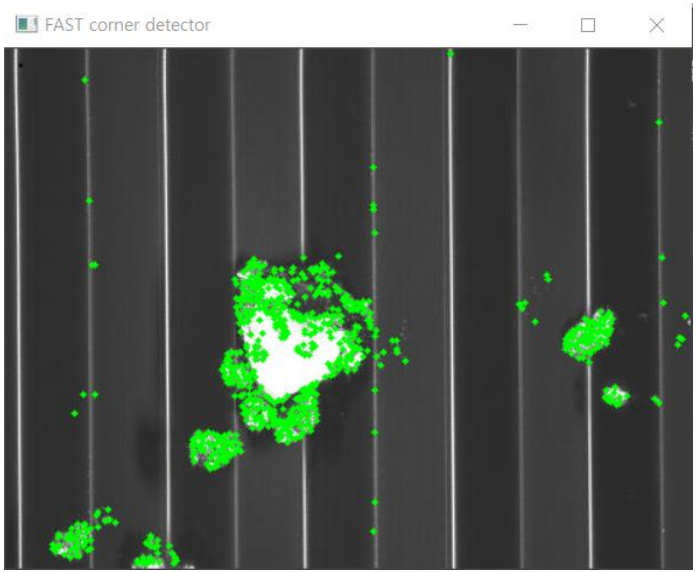
1. OK 제품- 실행 결과



추출 알고리즘 – Harris corner detector/FAST corner detector

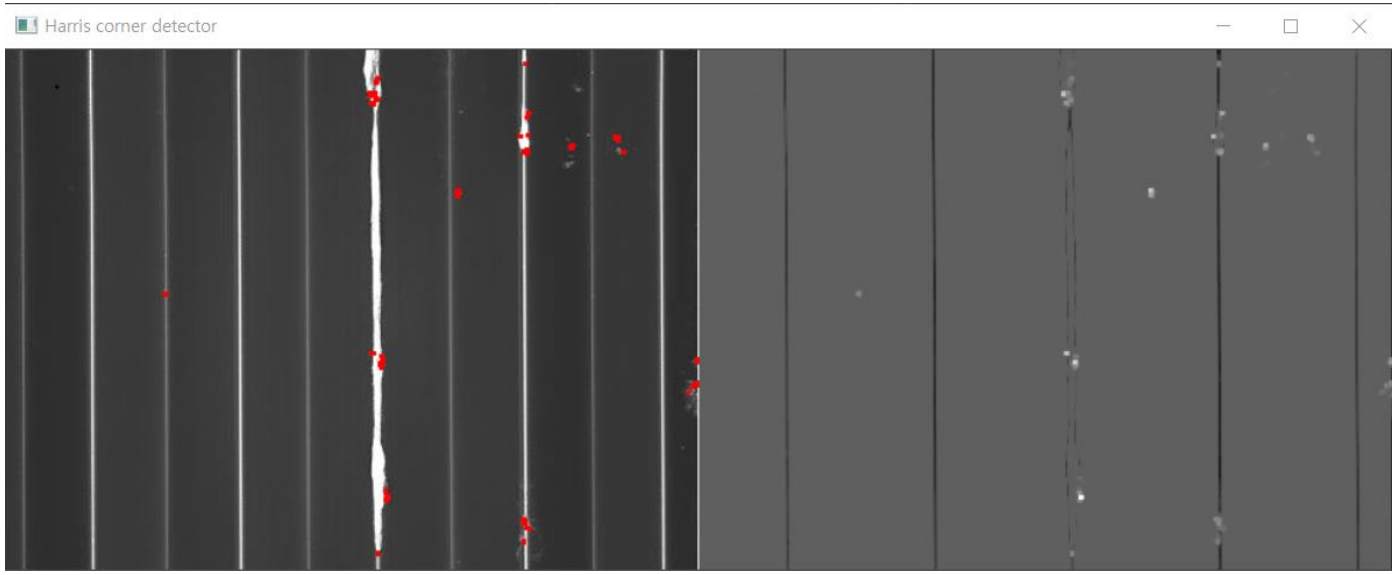
2. white spot – 실행결과

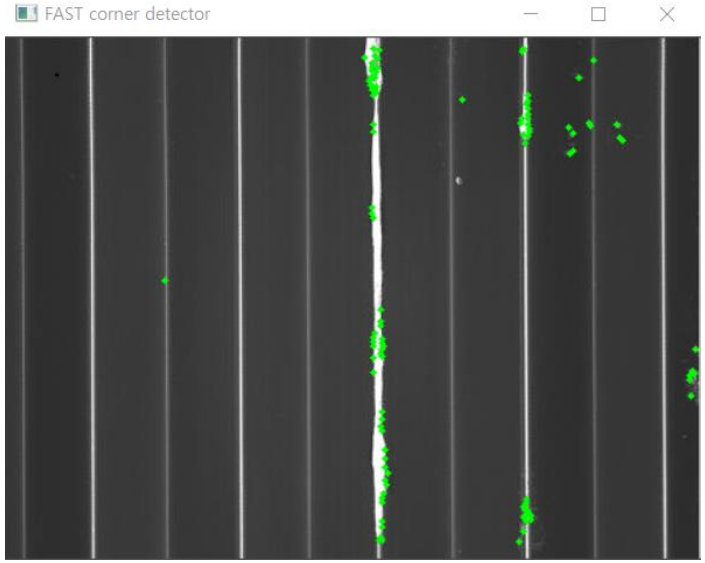




추출 알고리즘 – Harris corner detector/FAST corner detector

3. Scratch – 실행결과





실행 결과

- 당사 광학필름의 Prism 형상을 Harris corner detector 및 FAST corner detector 영상처리를 해 본 결과 OK품과 NG품을 확실히 구별 할 수 있었습니다.

감사합니다