3주차 과제 보고서

1. 이진화 및 히스토그램 분석

문제

텍스트, 폰트, 스크린샷, 대수학이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.  
  
텍스트, 스크린샷, 소프트웨어, 디스플레이이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.요구사항의 cv.threshold() 함수는 사용 방법이 ppt에 나와있다. 첫번째 인자에 흑백 전환된 이미지를 넣고 2번째 인자에 127을 사용하기만 하면 된다.

cv.calcHist() 함수도 제공이 된다.텍스트, 폰트, 멀티미디어 소프트웨어, 소프트웨어이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.  
수업 자료에 다음과 같이 히스토그램을 구하는 코드가 나와있으므로 그대로 활용하면 된다.

이 문제의 핵심은  
  
수업 자료에 포함되어 있지 않은 plt 함수들에 대해서 간략히 설명하면.  
plt.axis("off")  
이미지 표시용으로 눈금 및 축 제거  
plt.subplot(2, 3, 6)

전체 그림을 2행 3열로 나누고, 6번째 위치에 그래프를 그림.

plt.plot(hist\_binary, color='black')

hist\_binary 데이터를 선 그래프로 그림.

선의 색은 검정색.

plt.title("Binary Histogram")

현재 서브플롯의 제목 설정.

plt.xlabel("Pixel Value")

X축 라벨 설정.

plt.ylabel("Frequency")

Y축 라벨 설정.

plt.tight\_layout()

여백 자동 조정

plt.show()

그래프 표시

텍스트, 스크린샷, 라인, 도표이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

kernel = cv.getStructuringElement(cv.MORPH\_RECT, (5, 5))  
cv.getStructuringElement로 첫번째 인자가 사각형 커널로 설정, 두번째 인자가 크기 설정을 5x5로 해주어서 커널을 생성하고   
cv.morphologyEx(binary, cv.MORPH\_CLOSE, kernel) 식으로 첫번째 인자가 적용시킬 이미지, 두번째 인자가 모폴로지 연산 종류, 세번째 인자가 적용시킬 때 사용할 커널임을 알면 힌트에 나온 값들을 돌려서 쓰기만 하면 된다.  
예상외의 복병은 마지막 요구사항인데 원본 이미지가 너무 크므로 힌트에서 제시한대로 가로로 이어붙이면 아주 거대한 모니터를 사용하지 않는 이상 cv.imshow로 출력이 정상적으로 안되는데 resized\_result = cv.resize(result, (int(cols \* 1/4), int(rows \* 1/4))) 식으로 사이즈를 리사이징해서 출력해서 해결했다. (hstack을 사용하는 것은 요구 사항이 아닌 힌트에 불가하므로 np.vstack을 사용해서 세로로 붙이는것도 가능한 해결 방안이다. 이미지 출력에 다른 라이브러리를 쓰는 것은 근본적으로 이미지를 자동으로 리사이징 하느냐 마느냐 차이이므로 사용하지 않았다.)

  
3. 기하 연산 및 선형 보간 적용하기

텍스트, 스크린샷, 폰트, 문서이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.  
문제 해결의 핵심은 힌트가 함정을 내포하는 것을 알아채는것으로 docs 문서를 열어보면 쉽게 알 수 있다. ([OpenCV: Affine Transformations](https://docs.opencv.org/3.4/d4/d61/tutorial_warp_affine.html))  
cv.getRotationMatrix2D(center, 45, 1.5)는 첫번째 인자가 회전 중심으로 cols/2 rows/2 를 사용한다, 그 뒤는 각각 각도와 확대 크기로 45도 회전 및 1.5배 크기를 사용한다.

cv.warpAffine(img, M, (new\_w, new\_h), flags=cv.INTER\_LINEAR)  
역시 출력 이미지 크기는 세번째 인자로 (int(cols \* 1.5), int(rows \* 1.5))으로 지정한다 네번째 인자는 보간법 지정으로 선형 보간법으로 지정, 첫번째 인자는 원본 이미지, 두번째 데이터는 기하 연산을 위한 행렬을 지정한다.  
마지막으로 이미지 크기 차이를 메우기 위해 cv.copyMakeBorder(img, 0, int(rows \* 1.5) - rows, 0, int(cols \* 1.5) - cols, borderType=cv.BORDER\_CONSTANT, value=[0, 0, 0])

를 사용했다. 각각이 원본 이미지와 원본 이미지의 왼쪽으로 패딩할 값, 오른쪽으로 패딩 할 값, 위로 패딩할 값, 아래로 패딩할 값, border의 종류 (제로 패딩을 위해 cv.BORDER\_CONSTANT (상수 값으로 남은 영역을 처리)했다.) 패딩할 BGR값으로 오른쪽의 이미지와 크기를 맞추었다.

스크린샷, 나무이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.