컴퓨터비전 과제 2 리포트

2022-2학기

교과목: 컴퓨터비전

교과목교수: 송현주 교수님

소속: 컴퓨터학부

학번: 20170819

제출자: 정준영



목차

#1 사각형 내의 원의 수 세기 - 해결

#2 사각형 내의 원의 수 세기 - 해결

#3 주사위 읽기 (1) - 해결

#4 주사위 읽기 (2) - 해결

#5 주사위 읽기 (3) - 해결

**추가 공지 (Update 12/02)**

* #1과 #2에 대해서는 회전이 되지 않은 입력 이미지를 가정하고 작업하셔도 무방합니다.
* #3의 경우 밝은 배경의 이미지 하나 이상, 어두운 배경의 이미지 하나 이상에 대해서 잘 동작하면 됩니다.
* #4의 경우 추가로 올려드린 테스트 이미지들 중 하나 이상에 대해서 잘 동작하면 됩니다.
* #5의 경우 추가로 올려드린 테스트 이미지들 중 하나 이상에 대해서 잘 동작하면 됩니다.
* 잘 동작하는 이미지들에 대해서, 어떤 경우에 어떻게 처리하여 눈금을 읽을 수 있도록 했는지 보고서에 작성해주세요.

#1 사각형 내의 원의 수 세기

1. 문제 : 한 이미지 내에 하나의 사각형이 있고, 그 안에 n개의 원이 있을 때, 원의 숫자를 console에 출력하기
2. 조건 :

* Grayscale 입력
* 배경은 어두운 색으로 가정
* 사각형의 배경색은 밝은 색으로 가정
* 원의 색은 어두운 색으로 가정

1. 풀이

텍스트이(가) 표시된 사진

자동 생성된 설명

Imread()함수를 통해 이미지를 Grayscale로 입력 받습니다. 이후 이미지가 입력되지 않는 경우에 대해 에러를 처리합니다.

medianBlur()함수를 통해 약간의 블러링 효과를 주었습니다. medianBlur()는 소금&후추 영상에서 잡음 제거에 효과가 있기 때문에 그레이스케일 영상에서의 약간의 잡음들을 잘 잡아 줄 수 있지 않을까 생각해 사용했습니다. 이후 허프변환 원 검출 함수를 통해 원을 검출했습니다. 인자로 그레이스케일 영상을 넣어 주었고, 각각의 파라메터 값을 조정해가며 넣어주었습니다. 과제를 하며 추가로 인자에 대해 자세히 공부하면서 이를 활용해 과제에 적용했습니다.

dp:이미지 해상도, 값이 1이면 원본해상도. 값이 2이면 절반 해상도.

min\_dist: 검출할 원의 최소거리 – 원의 반지름이 아닌, 원의 밀도. 원이 오밀조밀하게 모여있으면 이값이 작아야 검출. 원이 넓게 골고루 퍼져있다면 오검출 되지 않게 적당히 조절할 필요 있음

parameter1: Canny edge detecion에서 높은 threshold 값. 작은값은 자동으로 0.5배로 적용됨.

Parameter2: 원 검출을 위한 정보. Accmulator의 threshold 값. – 너무 작으면 거짓 원이 검출.

실제 예제 이미지에 대해서 테스트 해본 결과, 해당 파라메터 값을 적용했을 때 가장 좋은 검출 성능을 보여주었습니다.

HoughCircles()함수 결과 리턴값에 원의 개수가 담기고, 이를 console창에 출력해 주었습니다.

1. 사용 이미지 및 결과



* 과제 링크에 첨부된 사이트에서 가져온 이미지를 테스트에 사용.

텍스트이(가) 표시된 사진

자동 생성된 설명

콘솔창에 결과로 “4”가 잘 나오는 것을 확인할 수 있습니다.

#2 사각형 내의 원의 수 세기

1. 문제 : 한 이미지 내에 여러 개의 사각형 존재. 각 사각형 안에 각각 n개의 원이 있을 때, 각 사각형 별로 내부에 있는 원의 숫자를 오름차순으로 console에 출력하기
2. 조건 :

* Grayscale 입력
* 배경은 어두운 색으로 가정
* 사각형의 배경색은 밝은 색으로 가정
* 원의 색은 어두운 색으로 가정

1. 풀이

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

함수 호출 순서

Contours\_basic()호출 -> if vtc == 4: (사각형 판별이 될 때(즉, 주사위)) -> img\_trim() – 해당 주사위를 잘라오는 함수 -> 자른 이미지를 갖고 hough\_circles()호출 -> 사각형 안에서 원의 개수를 세서 rst 배열에 저장.

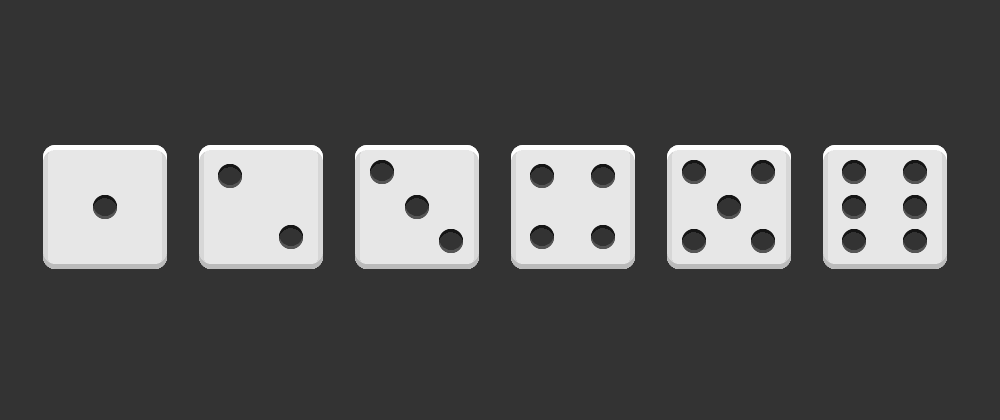
먼저 주사위 별로 원의 개수를 세기 위해서 주사위를 판단해야 했습니다. 그래서 사각형을 인식하기 위해 이미지를 읽은 후, threshold()를 통해 이미지를 이진화 해주었습니다. 이후에 findCountours()함수를 통해 통해 외곽선 계층정보를 담아왔고, 각각의 객체들을 판별해 주었습니다. 객체 크기를 400보다 작은것은 무시했고, 이때 문제가 없기에 설정했습니다.

이후에는 approxPolyDP()함수를 통해 근사화 작업을 진행했고, 이를 통해 점의 개수를 측정했습니다. 이때 점이 4개라는 것은 사각형이고 원하는 주사위 전면 모양이기에 해당 좌표를 추출하는 작업을 진행했습니다.

해당 좌표값을 갖고 img\_trim()함수를 통해 추출하고, 해당 이미지를 파라메터로 hough\_corcles()함수를 호출해서 해당 이미지(주사위)에 원이 몇개 있는지 세서 rst 결과 배열에 저장해주었습니다.

모든 작업이 끝난 이후에는 rst배열을 정렬해서 오름차순으로 출력했습니다.

1. 사용 이미지 및 결과



* 과제 링크에 첨부된 사이트에서 가져온 이미지를 테스트에 사용.

결과

텍스트이(가) 표시된 사진

자동 생성된 설명

주사위 마다 주사위 점의 개수를 잘 측정해서 오름차순으로 출력됨을 확인할 수 있습니다.

#3 주사위 읽기 (1)

1. 문제 : 책상 위에 놓인 임의의 주사위의 눈을 읽기. 한 이미지 내에 여러 개의 주사위 존재. 주사위의 눈을 오름차순으로 console에 출력하기
2. 조건 :

* Color 입력
* 주사위의 색상은 밝은 색 주사위 눈의 색상은 어두운 색
* 배경은 임의의 색 – 밝은 배경 혹은 어두운 배경

1. 풀이

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

2번과 다른 점은 컬러 사진이라는 점이었습니다. 그래서 이미지를 컬러로 일단 읽고, 그레이스케일로 변환하는 과정을 거쳐야 했습니다. 컬러 이미지이기에 조명이나 기타 다른 배경의 개입을 줄이기 위해서 노이즈를 주고, 가우시안블러를 적용했습니다.

이후 이미지를 이진화 하는 작업을 거쳤고, 똑같이 주사위로 판별이 되면 해당 주사위 사진을 잘라내어 해당 이미지에서 원을 검출하는 함수인 hough\_circles()에 인자로 넣어주어 주사위 마다 주사위 점의 개수를 파악했습니다.

이때 hough\_circles()와 contours\_basic()의 하이퍼파라메터들을 약간씩 조정해주었습니다.

‘

1. 사용 이미지 및 결과

밝은 배경 이미지 3-2

원격, 컨트롤러, 게임, 표면이(가) 표시된 사진

자동 생성된 설명

밝은 배경 이미지 3-2 결과

텍스트이(가) 표시된 사진

자동 생성된 설명

어두운 배경 이미지 3-4

컨트롤러, 원격, 게임, 다른이(가) 표시된 사진

자동 생성된 설명

어두운 배경 이미지 3-4 결과

텍스트이(가) 표시된 사진

자동 생성된 설명

#4 주사위 읽기 (2)

1. 문제 : 책상 위에 놓인 임의의 주사위의 눈을 읽기. 한 이미지 내에 여러 개의 주사위 존재. 주사위의 눈을 오름차순으로 console에 출력하기
2. 조건 :

* Color 입력
* 주사위 및 주사위 눈의 색상은 임의로 선정
* 어두운 주사위에 밝은 주사위 눈도 존재, 배경도 임의의 색

1. 풀이

해당 문제는 주사위와 주사위의 눈의 색상이 임의로 선정 되는 것 외에는 3번과 동일한 문제라고 생각했습니다. 그래서 컬러 이미지를 이진화 하는 작업에 초점을 맞추었습니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

주요 코드입니다. 색상을 스플릿 해가면서 여러 결과를 확인해 본 결과 temp변수에 담긴 결과가 이진화 했을 때 가장 구분이 잘 갔기에 해당 이미지를 선택했습니다. 이후 나머지의 로직은 2,3번과 동일하고, 하이퍼 파라메터 값만 조정해주었습니다. 그렇게 주사위를 찾고, 안에서 주사위의 점의 개수를 측정해주었습니다.

컬러 분배를 잘 못해서 일단 1개의 이미지만 동작하는것으로 판단이 되고, 후에 해당 부분을 보완하면 나머지 이미지에 대한 결과도 잘 나올 것으로 예상됩니다.

1. 사용 이미지 및 결과

예제 이미지 4-1에 동작

실내이(가) 표시된 사진

자동 생성된 설명

예제 이미지 4-1 동작 결과

텍스트이(가) 표시된 사진

자동 생성된 설명

#5 주사위 읽기 (3)

1. 문제 : 책상 위에 놓인 하나의 주사위의 눈을 읽기. 여러 면이 보이는 주사위에 대해서 가장 넓은 면의 주사위의 눈을 읽어서 console에 출력하기
2. 조건 :

* Color 입력
* 주사위의 색상은 밝은 색 주사위 눈의 색상은 어두운 색

1. 풀이

텍스트이(가) 표시된 사진

자동 생성된 설명

가장 고민이 많았던 문제였습니다. 일단 이진연산을 수행해 주었고, 여러 면이 보일 때, 가장 넓은 면적이 가장 밝다는 점을 이용해서 침식과 팽창연산을 통해 해당 부분을 강조 효과를 주었습니다. 이후에는 findContours를 활용해 원의 개수를 세주었는데 이렇게 하니 가장 넓은 면적의 주사위 점이 살아남아서 카운트 할 수 있었습니다. 밝기 차이가 아닌 외각선을 다르게 검출했다면 다른 사진들도 잘 찾을 방향이 있을것 같습니다.

1. 사용 이미지 및 결과

테스트에 통과한 5-5,5-6,5-8,5-11의 예제 사진입니다.



순서대로 5-5, 5-6, 5-8, 5-11의 실행 결과입니다.

텍스트이(가) 표시된 사진

자동 생성된 설명