OpenCV day1 hw1 결과보고서

2025407012/로봇학부/송연우

목차

1. smoothing image

픽셀들로 이루어진 이미지를 블러 처리하여 흐리게 만드는 이유는 블러 처리된 이미지에서 검출하려는 물체의 경계선의 두드러짐이 줄어들고 노이즈가 약해지기 때문입니다. 필터를 통해 이미지에 블러 처리를 하게 되면 이미지의 주요 내용은 그대로 유지하면서도 세부적인 디테일이 덜 강조되어 오류를 줄일 수 있습니다. Hw1 과제에서는 대표적인 필터 가우시안 블러를 사용했습니다. opencv 에서 자체적으로 제공하는 이미지 흐림 함수는 4가지 정도가 있는데 각각 boxFilter(), GaussianBlur(), medianBlur(), bilateralFilter()가 있습니다. 가우시안 필터를 사용한 이유는, 해당 필터의 적용 과정에서 커널의 중심에서 멀어질수록 가중치를 감소시키며 픽셀에 변화를 주어변화가 부드럽게 나타나기 때문이며, 이 때문에 물체 경계선을 부드럽게 하는 데 효과적일 것이라 생각하여 사용하게 되었습니다.

2. hw1.hpp

클래스 선언부, #include 코드 작성부입니다. cpp 에서 사용할 클래스로 따로 클래스를 상속하지는 않는 HW1클래스를 생성했으며 public 부분에는 생성자 HW1() 과 void 형 make_binary(), showimage()를 선언했습니다. 생성자에서는 각각의 색상을 추출할 때 사용할 upper, lower 변수를 설정했으며 make_binary 에서는 필터 처리 하지 않은 이미지/처리한 이미지를 이진화한 이미지를 생성하고, showimage 에서는 imshow 로 생성한 이미지를 화면에 출력했습니다.

3. hw1.cpp

(1) 먼저 생성자 함수에 대한 정의 부분입니다.

```
img = imread("/home/yu/colcon_ws/src/cv_hw1/homework1.jpg");
```

우선 Mat img 등등 사용할 변수나 객체들을 hpp 파일에서 선언했으므로 imread 로 해당 절대경로에 위치한 이미지 파일을 읽어와 img 에 저장합니다.

```
cvtColor(img, img_hsv, COLOR_BGR2HSV); //채널 변환
```

jpg 이미지는 rgb 값을 색상으로 사용하므로 cvtcolor() 함수의 세 번째 인자에 색 표현 방식을 rgb 에서 HSV 로 변환합니다.

```
GaussianBlur(img, img_b, Size( ), (double)3);
```

필터 중 가우시안 블러를 적용해 img_b 에 저장합니다. 여기서 rgb 인 img 에 블러 처리를 하고 아래 사진처럼 hsv 사진에 다시 블러 처리를 하는 이유는 이후에 imshow 로 img 와 img_b 를 출력하여

블러처리한 이미지와 아닌 이미지를 대조비교하기 위해서입니다. 그리고 함수의 인수 설정으로는 아래 이미지와 같이 가장 간단하게 사용할 수 있는 함수 형태로 사용해 보았습니다.

```
cv::GaussianBlur(input_img, output_img, Size(size, size), sigma_value);
```

인풋/아웃풋 이미지 외의 인수에 대해서는 sigma_value 를 3으로 설정하고 size 를 공란으로 비워 두었는데, sigma_value 를 1이나 2로 두었을 때는 블러 효과를 준 차이가 크게 보이지 않았고 3을 초과한 값으로 설정했을 때는 너무 흐려져서 이진화 추출에 문제가 생겼기 때문에 3으로 설정하게 되었습니다. size 값을 비워 둔 이유는 표준 편차로부터 커널 크기를 자동으로 결정되게 하기 위함입니다.

```
GaussianBlur(img_hsv, img_blured, Size( ), (double)3);
```

다시 돌아와서, inRange()에 사용할 입력 이미지는 img_blured 와 img_hsv 입니다.

```
lower_red_1 = Scalar(0, 50, 50);
```

다음으로 inRange()함수에 매개변수로 들어갈, 이진화에서 빨간색 공을 검출할 때 1로 표시될 가장 낮은 값을 설정합니다.

```
upper_red_1 = Scalar(10, 255, 255);
```

가장 높은 임계값도 설정합니다. red 뒤에 _1이 붙는 이유는 red 의 Scalar()함수에 들어가는 값들을 조정했을 때 구간을 2개로 나누지 않으면 뒤의 배경색까지 1로 표시되는 현상이 나타났기 때문입니다.

```
lower_green = Scalar(35, 50, 50);
```

red 에 대해 설정을 마치면 green 과 blue 에 대해서도 동일하게 설정합니다. 이 둘은 red 와는 다르게 값의 범위를 쪼개지 않아도 정상적으로 검출되었습니다.

(2) make_binary()함수입니다.

```
inRange(img_hsv, lower_red_1, upper_red_1, red_mask);
```

inRange 함수를 사용해 red_mask 에 이진화된 이미지를 저장합니다.

```
inRange(img_hsv, lower_red_2, upper_red_2, red_image);
```

다른 색과 달리 red 의 경우에는 Scalar 인자값이 두 종류이므로 이진화된 이미지를 서로 다른 이미지에 저장 후 imshow 함수에서 더해 합친 이미지를 결과값으로 출력했습니다.

(3) showimage()함수입니다.

```
imshow("img", img);
```

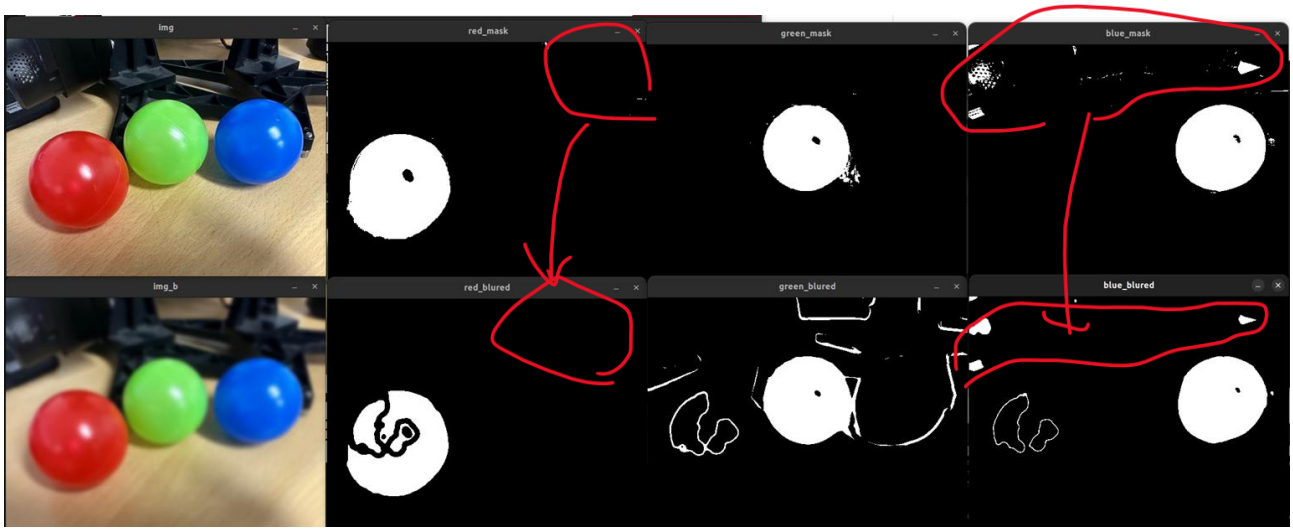
```
imshow("img_b", img_b);
```

원본 이미지와 가우시안 블러 처리된 이미지를 함께 띄워 줍니다.

```
imshow("red_mask", red_image + red_mask);
```

red 이미지의 경우 두 이미지를 합쳐 출력하고, 나머지는 정상적으로 출력합니다.
inary
(4) main 함수에서의 실행 순서는 HW1객체 생성->make_binary()->showimage()순입니다.

4. 실행 결과



위쪽부분의 이미지가 원본, 아래쪽이 블러 처리된 이미지입니다. 가우시안 블러를 적용 후 색상 추출에 변화가 생기기는 했지만 빨간색으로 동그라미 친 부분과 같이 노이즈가 줄어들고 물체의 경계선이 필터 처리 전보다 부드러워졌음을 알 수 있었습니다.