Color Detection Algorithm

PROTOTYPING

해당 알고리즘은 **시범 프로그래밍**으로 전체적인 동작을 빠르게 구현하고 시각화한 것입니다

각 이미지는 이해를 돕기 위한 예시 사진입니다

MADE BY HWANG WAS COMPLETED
02,02,2024

Use SW

Tool - Visual Studio 2022

Languages - C ++

Library - OpenCV

Need HW – Webcam or USB Camera

```cpp
// 카메라 열기
VideoCapture cap(0);

if (!cap.isOpened()) {
    cout << "Error opening camera." << endl;
    return -1;
}

bool motorOn = false;  // 모터 상태

while (true) {
    Mat frame;
    cap >> frame; // 프레임 읽기

    if (frame.empty()) {
        cout << "Empty frame." << endl;
        break;
    }
```

1. 카메라 영상 처리

- 'VideoCapture' 클래스를 통해 카메라를 읽음
- 'Mat frame' 카메라의 프레임(frame) 값을 저장하고 읽어 들이기 위한 행렬 클래스
- '!cap.isOpened()' 카메라를 읽을 수 없을 때
- 'frame.empty()' 프레임 값이 비어 있거나 없을 경우

2. 색상 공간 변환

```
// BGR 이미지를 HSV 이미지로 변환
Mat hsv;
cvtColor(frame, hsv, COLOR_BGR2HSV);
```

- BGR 이미지를 HSV 이미지로 변환하는 과정
- 색상 H(Hue), 색의 종류, 빨강, 파랑, 녹색등의 다양한 색상
- 채도 S(Saturation), 색의 강도 또는 순수성, 채도가 높을수록 색은 더 강렬하고 순수하며 회색 음영으로 갈수록 채도는 감소
- 명도 V(Value), 색의 밝기, 명도가 높을 수록 더 밝고, 낮을 수록 어두워 짐
- RGB로는 색상이 점진적으로 바뀌는 기울기를 구현하기가 힘들지만 이 HSV를 사용하면 두 개의 변수를 고정하고 하나의 변수만 움직이는 방법으로 음영변화, 채도변화 등의 표현이 쉬워짐

3. 색상 범위 설정

```
// 특정 HSV 범위 설정 (녹색의 경우 예시)
Scalar lower_hsv = Scalar(60, 60, 60);  // 최소값
Scalar upper_hsv = Scalar(80, 255, 255); // 최대값
```

특정 HSV 범위를
설정하여 감지할 색상
지정

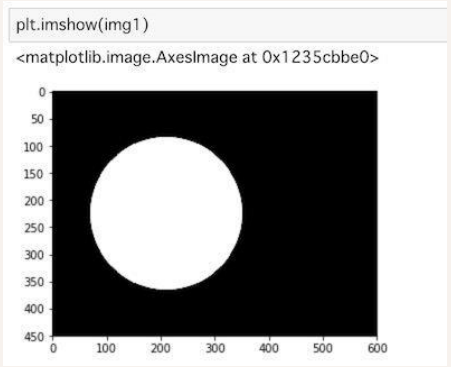
색상의 최소 범위 값과
최대 범위 값을
조정하여 감지하기 위한
특정 색상의 범위 지정

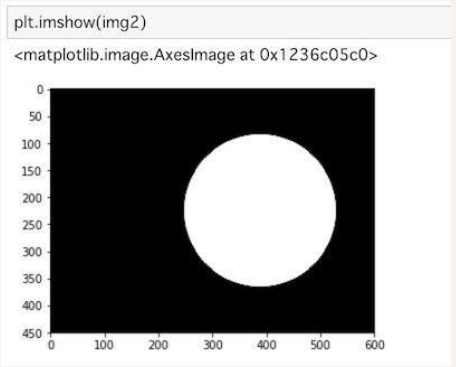
4. 이진 마스킹

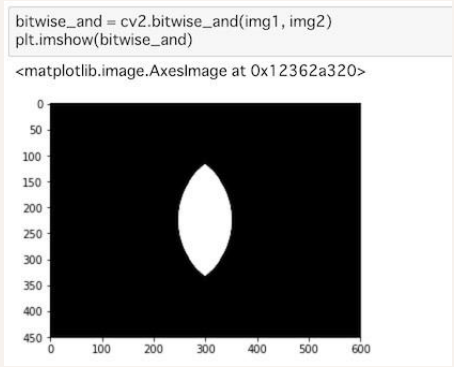
- 설정한 색상 범위에 해당하는 영역을 찾기 위한 ‘inRange’ 함수 사용 이진 마스크 생성

5. 영상 처리

```
// 녹색 이외의 부분을 검은색으로 처리
Mat result;
bitwise_and(frame, frame, result, mask);
```







- 'bitwise_and' 함수를 사용하여 마스크를 적용 녹색 이외의 부분을 검은색으로 만듦
- 두 행렬의 원소 간 혹은 행렬 원소와 스칼라 간의 비트별 논리곱(AND) 연산을 수행함을 의미
- 1번 이미지의 하얀 원의 영역과 2번 이미지의 하얀 원의 영역을 비교하여 두 이미지의 영역이 교집합이 되는 부분을 제외한 나머지 영역을 검은색으로 만듦

6. 영역 크기 측정

```
// 녹색 영역의 크기 계산
int greenAreaSize = countNonZero(mask);
```

'countNonZero' 함수를 사용하여 특정 색 영역의 크기를 계산함

이는 주어진 배열 또는 이미지에서 0이 아닌 값의 개수를 세는 함수

흑백 이미지에서 0이 아닌 픽셀의 개수를 계산

7. 모터 제어 결정

```
// 일정 크기 이상이면 "모터 on", 그렇지 않으면 "모터 off" 출력
if (greenAreaSize > 1000) {
    cout << "모터 on" << endl;
    motorOn = true;
}
else {
    if (motorOn) {
        cout << "모터 off" << endl;
        motorOn = false;
    }
}
```

- 특정 색상의 영역 크기를 기준으로 일정 크기 이상이면 모터를 작동 시키는 명령을 실행 그렇지 않을 시 모터의 작동을 중단하는 명령을 실행
- 해당 알고리즘에선 모터를 제어하는 명령이 아닌 모터의 상태를 출력하는 텍스트를 표시함

구현 결과
영상 이미지

해당 소스코드와 영상
이미지파일을 저장하지 못해 작업
촬영 이미지를 가져왔습니다