
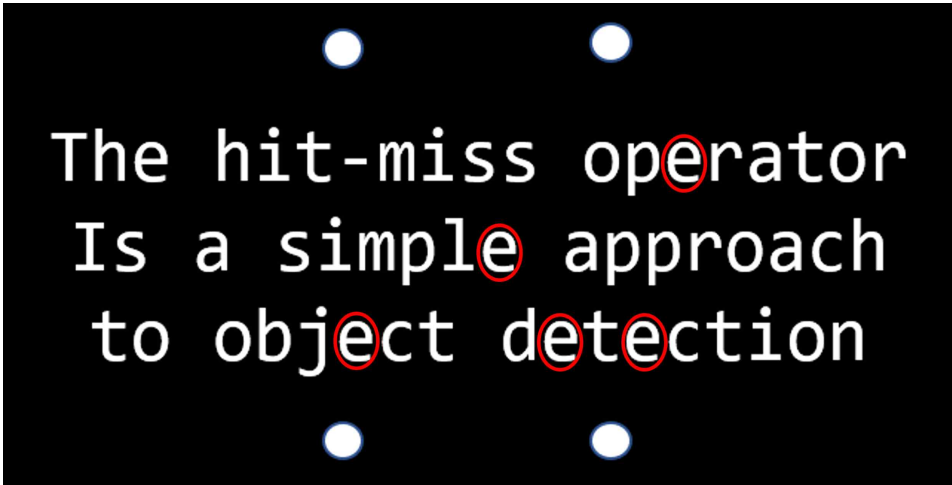


■ 과제 #3: 2023학년 1학기

교과목 / 담당교수	디지털영상처리 / 김남규	주차	14주차
이름	신민주	학번	20213067
■ 과제 (295페이지, 8번)			
문자들이 들어 있는 이진 영상에서 알파벳 “e”를 찾는 프로그램을 작성하라			
■ 입력 영상: HitMissInput.png			
			
■ 결과 영상: 아래 그림과 같이 “e”의 위치에 빨간 타원이 겹쳐 그려지도록 한다.			
			
■ 제출물: 구현 코드, 결과 화면, 고찰 내용 등을 작성하여 PDF 파일로 제출한다			

[구현 코드]

```
#include "opencv2/opencv.hpp"
#include <iostream>
using namespace cv;
using namespace std;
```

```

int main()
{
    // 이미지 파일 로드
    Mat inputImage = imread("d:/images/HitMissInput.png", IMREAD_GRAYSCALE);

    // 찾고자 하는 글자의 이미지 로드 및 이진화
    Mat templateImage = imread("d:/images/found_e.png", IMREAD_GRAYSCALE);
    Mat binaryTemplate;
    threshold(templateImage, binaryTemplate, 128, 255, THRESH_BINARY);

    // 구조 요소 생성
    Mat element = binaryTemplate.clone();

    // 팽창 연산 수행
    Mat dilated;
    dilate(inputImage, dilated, element);

    // 침식 연산 수행
    Mat eroded;
    erode(inputImage, eroded, element);

    // 찾은 문자 영역 추출
    Mat result;
    subtract(dilated, eroded, result);

    // 찾은 문자에 빨간 동그라미 표시
    vector<vector<Point>> contours;
    findContours(result, contours, RETR_EXTERNAL, CHAIN_APPROX_SIMPLE);

    Mat outputImage = imread("d:/images/HitMissInput.png");

    for (const auto& contour : contours) {
        Rect boundingRect = cv::boundingRect(contour);
        Mat roi = inputImage(boundingRect);

        // 템플릿 매칭 수행
        Mat matchResult;
        matchTemplate(roi, binaryTemplate, matchResult, TM_CCOEFF_NORMED);

        double minVal, maxVal;
        Point minLoc, maxLoc;

        while (true) {
            minMaxLoc(matchResult, &minVal, &maxVal, &minLoc, &maxLoc);

            // 매칭 결과를 이용하여 일치하는지 확인
            if (maxVal >= 0.8) {
                Point2f center(maxLoc.x + boundingRect.x + binaryTemplate.cols / 2,
                               maxLoc.y + boundingRect.y + binaryTemplate.rows / 2);
                circle(outputImage, center, binaryTemplate.cols / 2, Scalar(0, 0, 255), 2);

                // 이미지에서 찾은 영역을 검은색으로 채우기
            }
        }
    }
}

```

```

        rectangle(matchResult, Rect(maxLoc, binaryTemplate.size()), Scalar(0), FILLED);
    }
    else {
        break;
    }
}
}

// 결과 이미지 출력
imshow("Hit-or-Miss", outputImage);
waitKey(0);

return 0;
}

```

[결과 화면]



[고찰 내용]

문자 ‘e’를 찾기 위해 e의 구조 요소를 e의 문자만 잘라서 새로운 png 파일로 저장하고 그 파일을 이진화하여 구조 요소로 지정한다. 팽창과 침식 연산을 이용해 hit-or-miss 연산을 한 뒤 그 값을 이용해 찾은 문자 영역을 추출한다. 그 뒤 match 함수를 사용해서 구조 요소와 원본 영상에서 똑같은 문자를 찾는다. 찾은 문자를 빨간 동그라미로 찾기 위해 임계값 이상일 때만 나타나도록 표시하고, 이미지에서 찾은 값을 검은색으로 변경해주어야 한다. 만약, 해주지 않는다면 현재 원본 영상에 있는 모든 문자 ‘e’를 찾을 수 없게 될 것이다. 이번 장에서 모폴로지 연산을 통해 문자를 인식할 수 있는 방법에 대해 자세히 배우며, 더 나아가 영상을 찍고 해당하는 글자를 자동으로 복사해주는 것도 활용할 수 있겠다는 생각이 들었다.