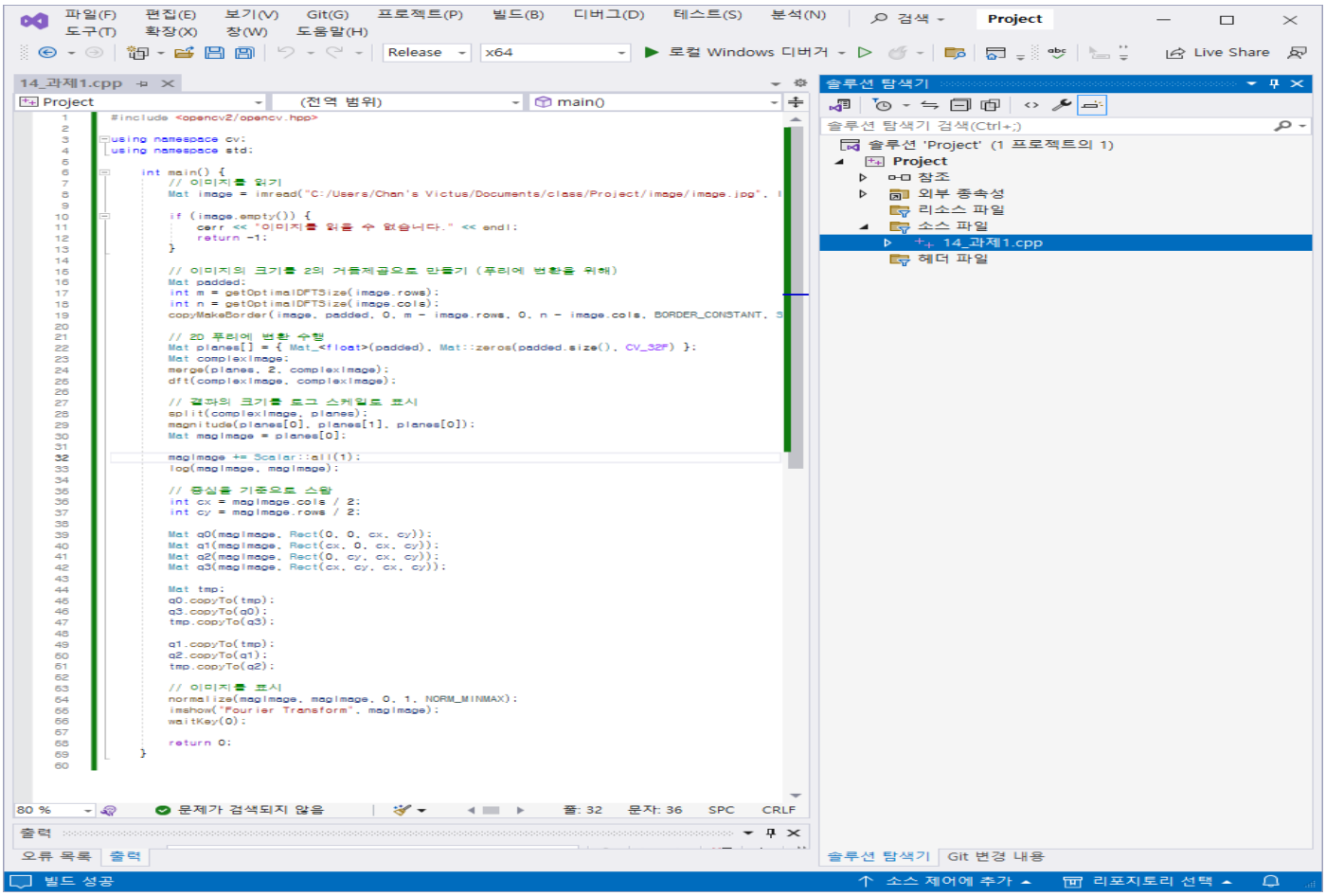
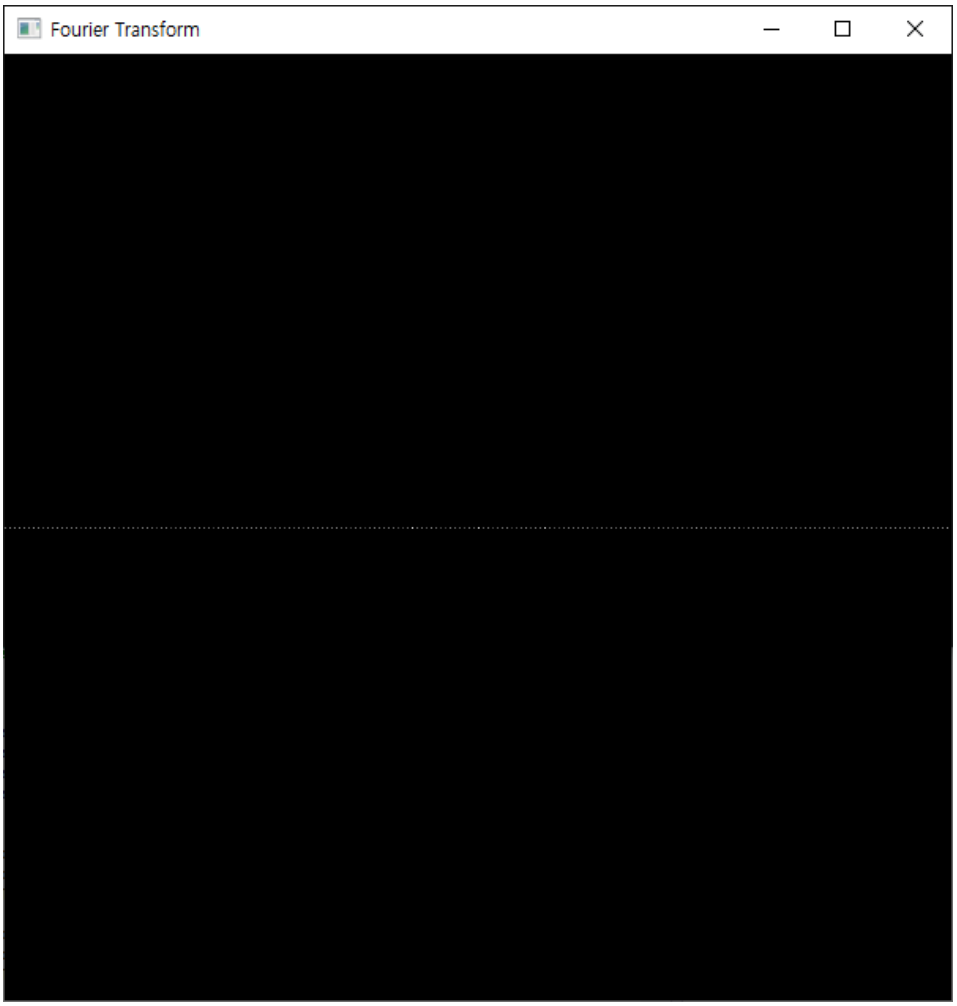
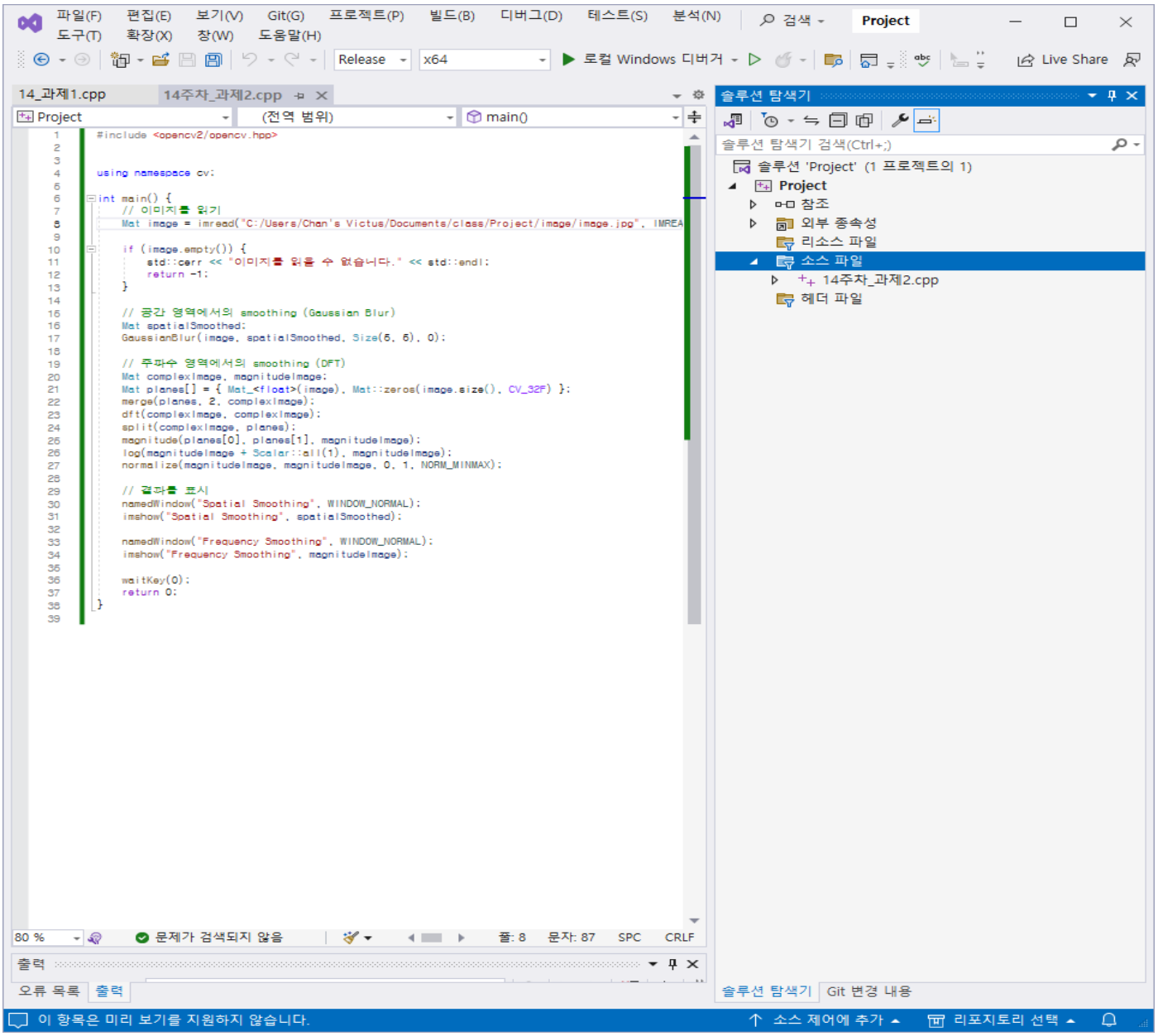
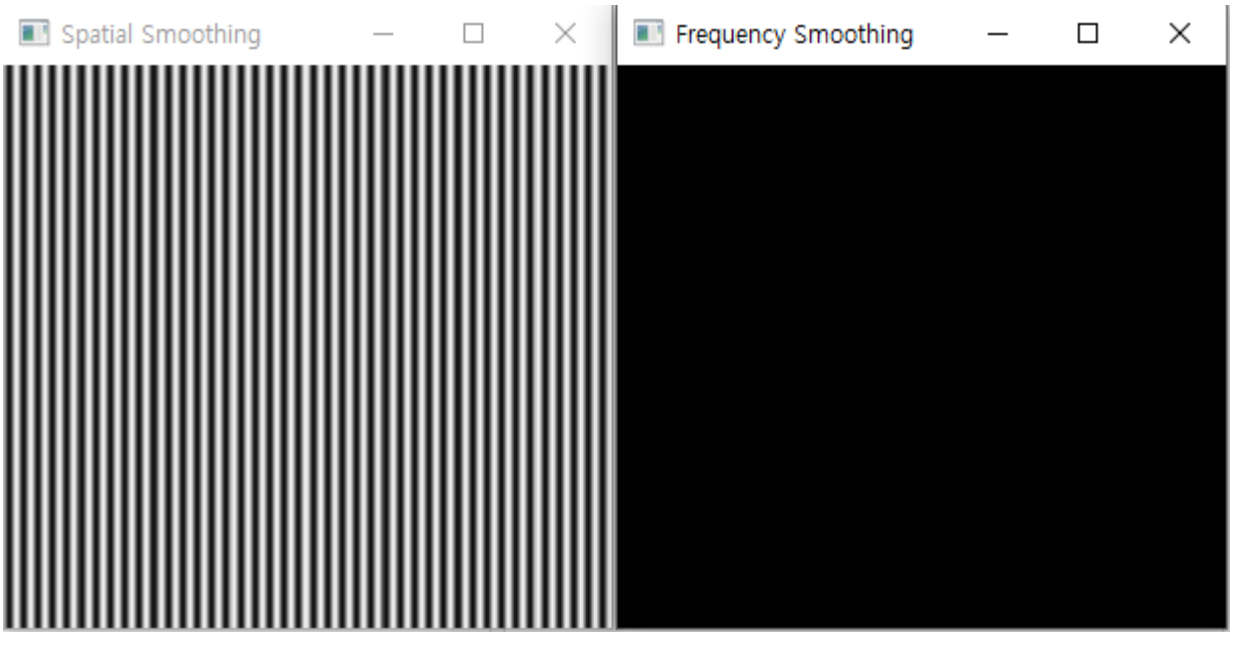
14-주파수영역 처리.pdf 1번





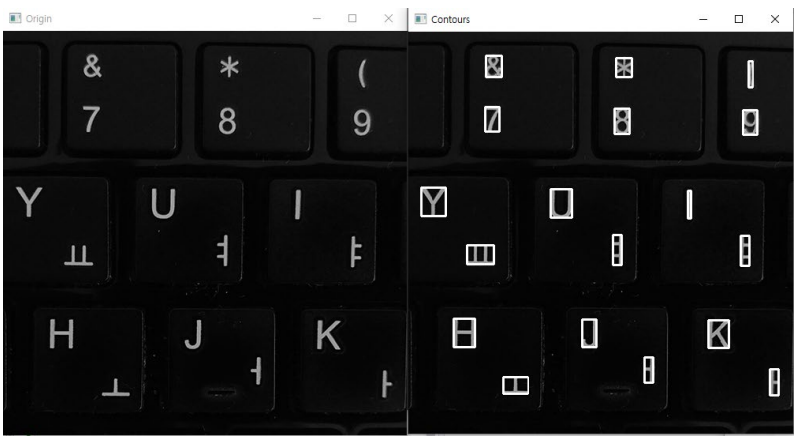
14-주파수영역 처리.pdf 2번





15-컬러영상처리.pdf 1번

```cpp
#include <opencv2/opencv.hpp>

using namespace cv;

int main() {
    // 이미지를 읽기
    Mat image = imread("C:/Users/Chan's Victus/Documents/class/Project/image/keyboard.bmp", IM

    if (image.empty()) {
        std::cerr << "이미지를 읽을 수 없습니다." << std::endl;
        return -1;
    }

    // 이미지 이진화
    Mat binaryImage;
    threshold(image, binaryImage, 128, 255, THRESH_BINARY);

    // 레이블링
    Mat labeledImage, stats, centroids;
    int nLabels = connectedComponentsWithStats(binaryImage, labeledImage, stats, centroids);

    // 원본 이미지에 컨투어 그리기
    Mat contourImage = image.clone();
    std::vector<std::vector<Point>> contours;
    findContours(binaryImage, contours, RETR_EXTERNAL, CHAIN_APPROX_SIMPLE);

    for (const auto& contour : contours) {
        Rect rect = boundingRect(contour);
        rectangle(contourImage, rect, Scalar(255, 0, 0), 2);
    }

    // 결과 표시

    imshow("Origin", image);

    namedWindow("Contours", WINDOW_NORMAL);
    imshow("Contours", contourImage);

    waitKey(0);

    return 0;
}
```



15-컬러영상처리.pdf 2번

```cpp
#include <opencv2/opencv.hpp>

using namespace cv;
using namespace std;

int main() {
    // 이미지를 읽기
    Mat image = imread("C:/Users/Chan's Victus/Documents/class/Project/image/shape.bmp");

    if (image.empty()) {
        cerr << "이미지를 읽을 수 없습니다." << endl;
        return -1;
    }

    // 그레이스케일로 변환
    Mat gray;
    cvtColor(image, gray, COLOR_BGR2GRAY);
    // 영상 반전
    Mat invertedImage = 255 - gray;


    // 이진화
    Mat binary;
    threshold(invertedImage, binary, 10, 255, THRESH_BINARY );

    //imshow("11", binary);

    // 모폴로지 연산을 통한 노이즈 제거
    Mat kernel = getStructuringElement(MORPH_RECT, Size(5, 5));
    morphologyEx(binary, binary, MORPH_CLOSE, kernel, Point(-1, -1), 1);

    // 컨투어 추출
    vector<vector<Point>> contours;
    findContours(binary, contours, RETR_LIST, CHAIN_APPROX_SIMPLE);

    // 그림 그리기
    Mat resultImage = image.clone();
    for (size_t i = 0; i < contours.size(); i++) {
        // 외곽선에 대한 근사 다각형 구하기
        vector<Point> approxCurve;
        approxPolyDP(contours[i], approxCurve, arcLength(contours[i], true) * 0.02, true);

        // 일정 크기 이상의 다각형만 그리기
        if (approxCurve.size() >= 3) {
            drawContours(resultImage, vector<vector<Point>>{contours[i]}, 0, Scalar(0, 0, 0),
        }
    }

    // 결과 표시

    imshow("Original Image", image);

    imshow("Result Image", resultImage);

    waitKey(0);

    return 0;
}
```