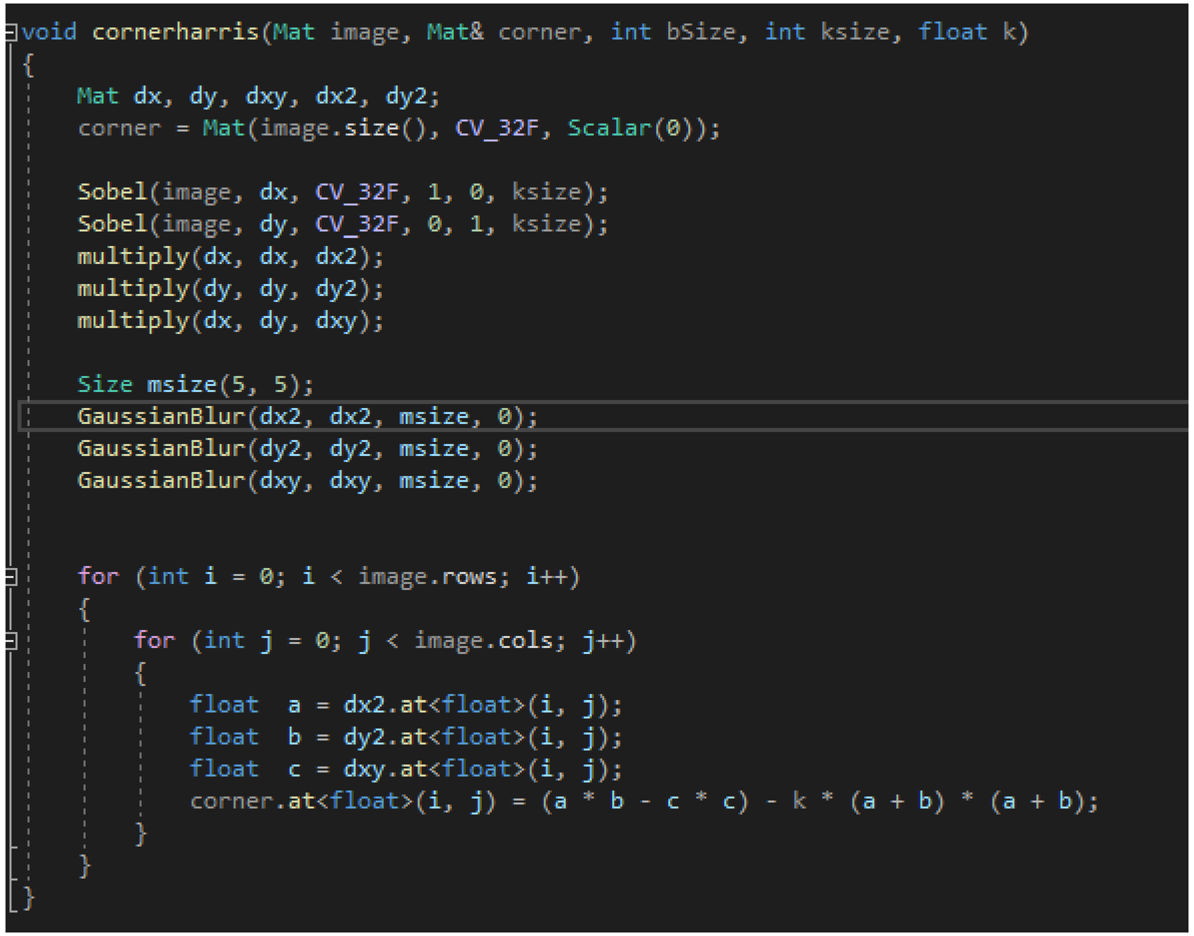
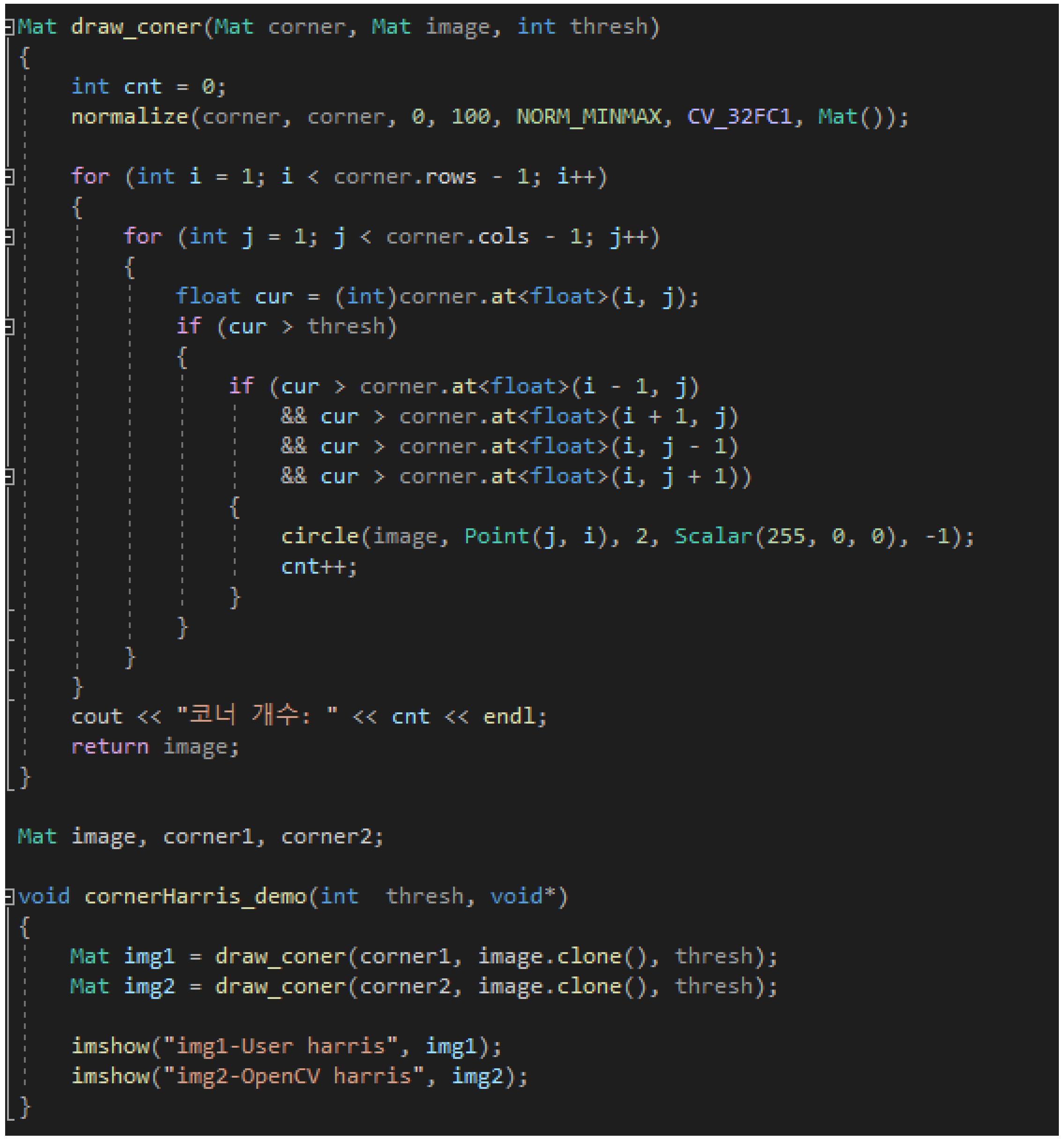
영상처리 실제 - 12주차 실습

2023254009 최현동

: 17.특징추출(2) – p.12 ~15

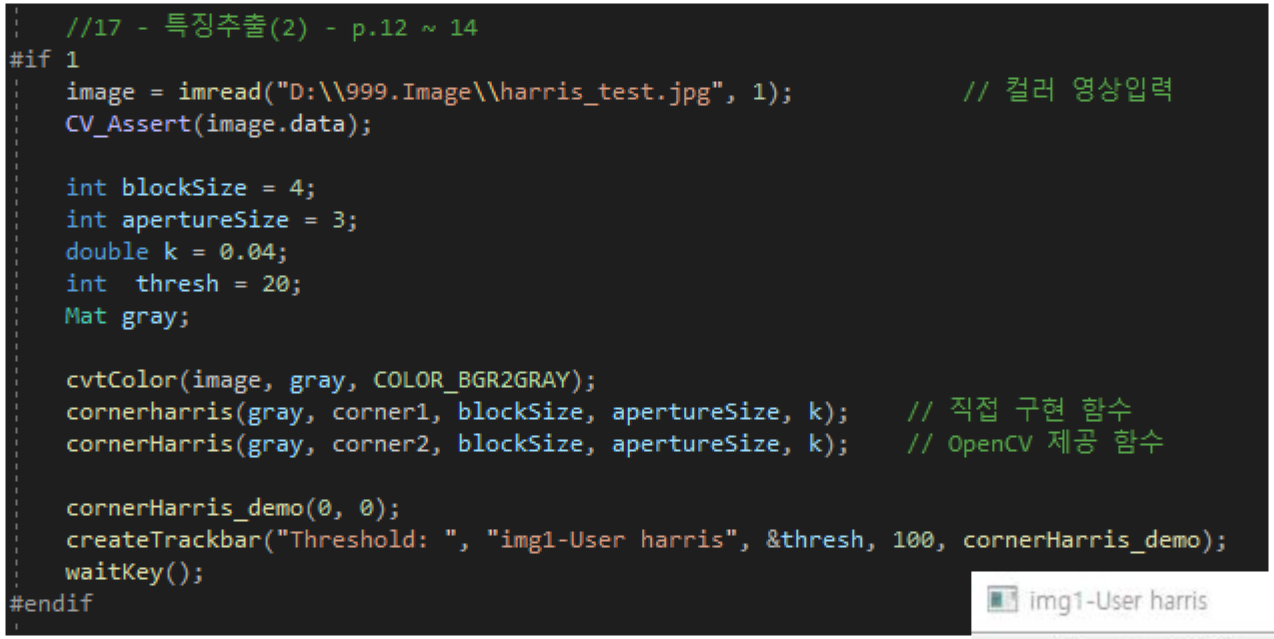


```
void cornerharris(Mat image, Mat& corner, int bSize, int ksize, float k)
{
    Mat dx, dy, dxy, dx2, dy2;
    corner = Mat(image.size(), CV_32F, Scalar(0));

    Sobel(image, dx, CV_32F, 1, 0, ksize);
    Sobel(image, dy, CV_32F, 0, 1, ksize);
    multiply(dx, dx, dx2);
    multiply(dy, dy, dy2);
    multiply(dx, dy, dxy);

    Size msize(5, 5);
    GaussianBlur(dx2, dx2, msize, 0);
    GaussianBlur(dy2, dy2, msize, 0);
    GaussianBlur(dxy, dxy, msize, 0);


    for (int i = 0; i < image.rows; i++)
    {
        for (int j = 0; j < image.cols; j++)
        {
            float  a = dx2.at<float>(i, j);
            float  b = dy2.at<float>(i, j);
            float  c = dxy.at<float>(i, j);
            corner.at<float>(i, j) = (a * b - c * c) - k * (a + b) * (a + b);
        }
    }
}
```

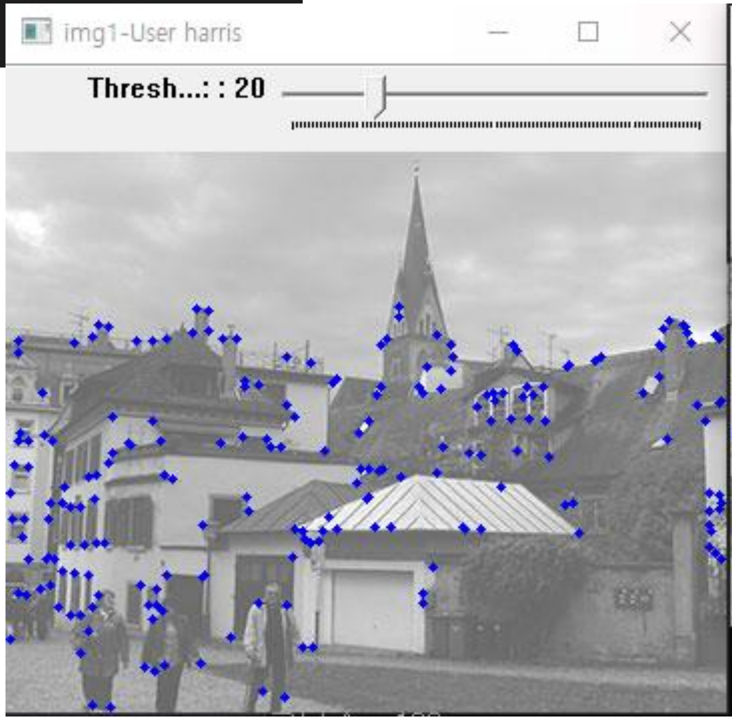


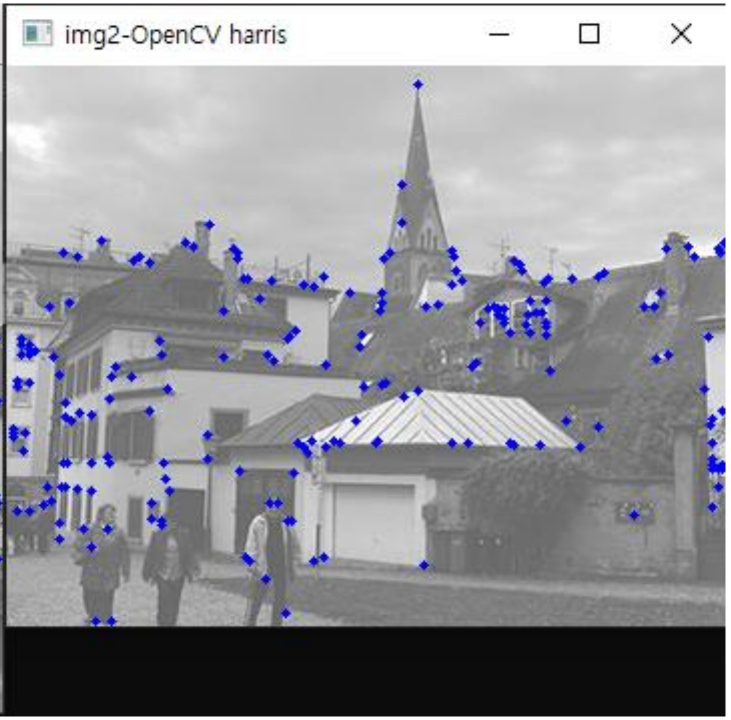
```
Mat draw_coner(Mat corner, Mat image, int thresh)
{
    int cnt = 0;
    normalize(corner, corner, 0, 100, NORM_MINMAX, CV_32FC1, Mat());

    for (int i = 1; i < corner.rows - 1; i++)
    {
        for (int j = 1; j < corner.cols - 1; j++)
        {
            float cur = (int)corner.at<float>(i, j);
            if (cur > thresh)
            {
                if (cur > corner.at<float>(i - 1, j)
                    && cur > corner.at<float>(i + 1, j)
                    && cur > corner.at<float>(i, j - 1)
                    && cur > corner.at<float>(i, j + 1))
                {
                    circle(image, Point(j, i), 2, Scalar(255, 0, 0), -1);
                    cnt++;
                }
            }
        }
    }
    cout << "코너 개수: " << cnt << endl;
    return image;
}

Mat image, corner1, corner2;

void cornerHarris_demo(int  thresh, void*)
{
    Mat img1 = draw_coner(corner1, image.clone(), thresh);
    Mat img2 = draw_coner(corner2, image.clone(), thresh);

    imshow("img1-User harris", img1);
    imshow("img2-OpenCV harris", img2);
}
```

: 17.특징추출(2) – p.12 ~15

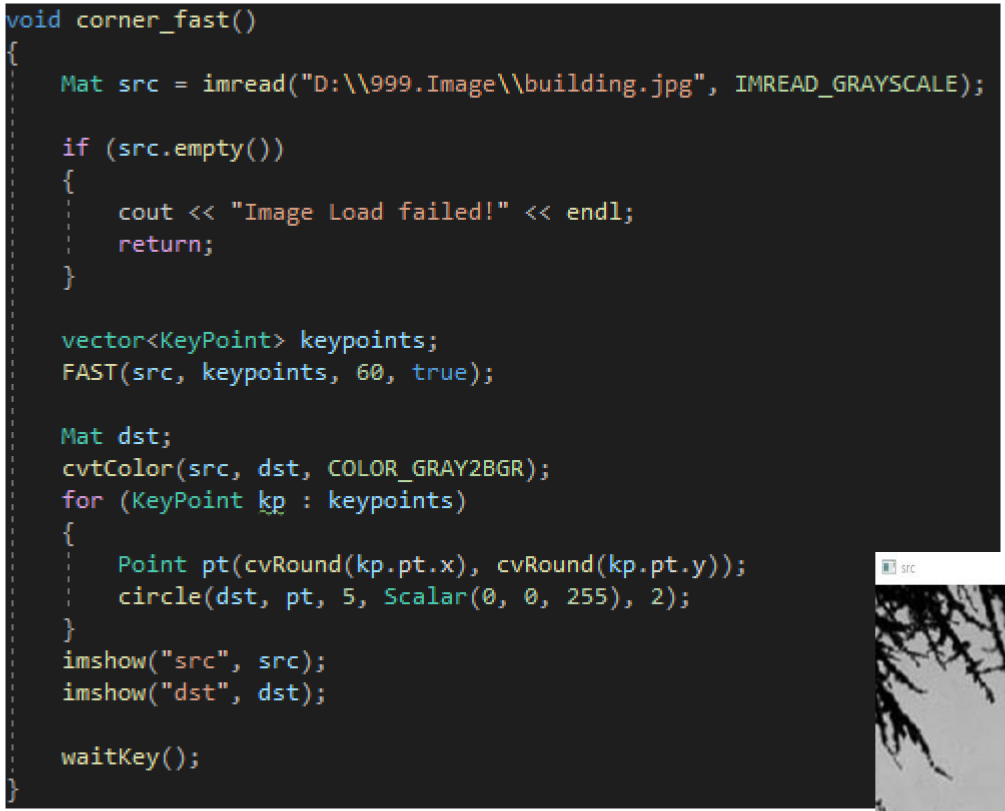


```
    //17 - 특징추출(2) - p.12 ~ 14
#if 1
    image = imread("D:\\999.Image\\harris_test.jpg", 1);            // 컬러 영상입력
    CV_Assert(image.data);

    int blockSize = 4;
    int apertureSize = 3;
    double k = 0.04;
    int  thresh = 20;
    Mat gray;

    cvtColor(image, gray, COLOR_BGR2GRAY);
    cornerharris(gray, corner1, blockSize, apertureSize, k);    // 직접 구현 함수
    cornerHarris(gray, corner2, blockSize, apertureSize, k);    // OpenCV 제공 함수

    cornerHarris_demo(0, 0);
    createTrackbar("Threshold: ", "img1-User harris", &thresh, 100, cornerHarris_demo);
    waitKey();
#endif
```

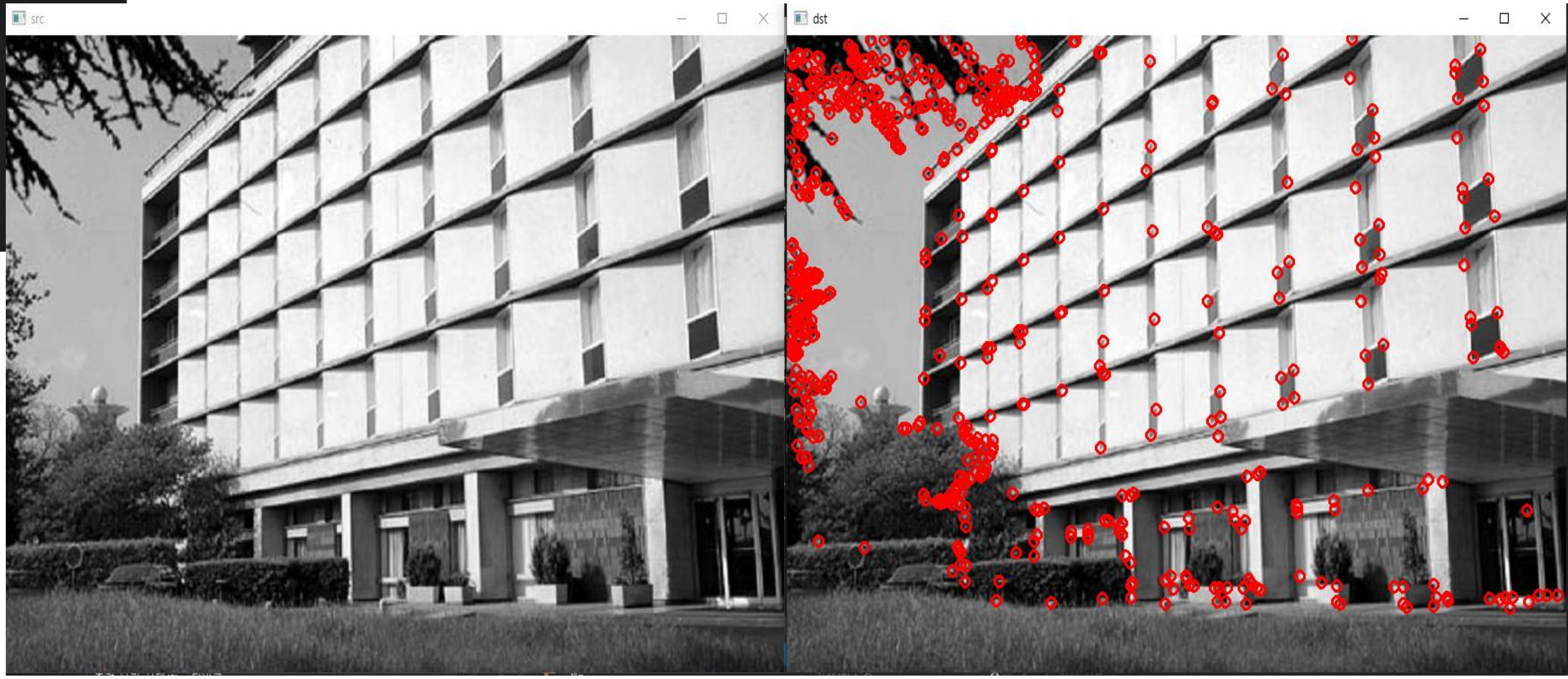




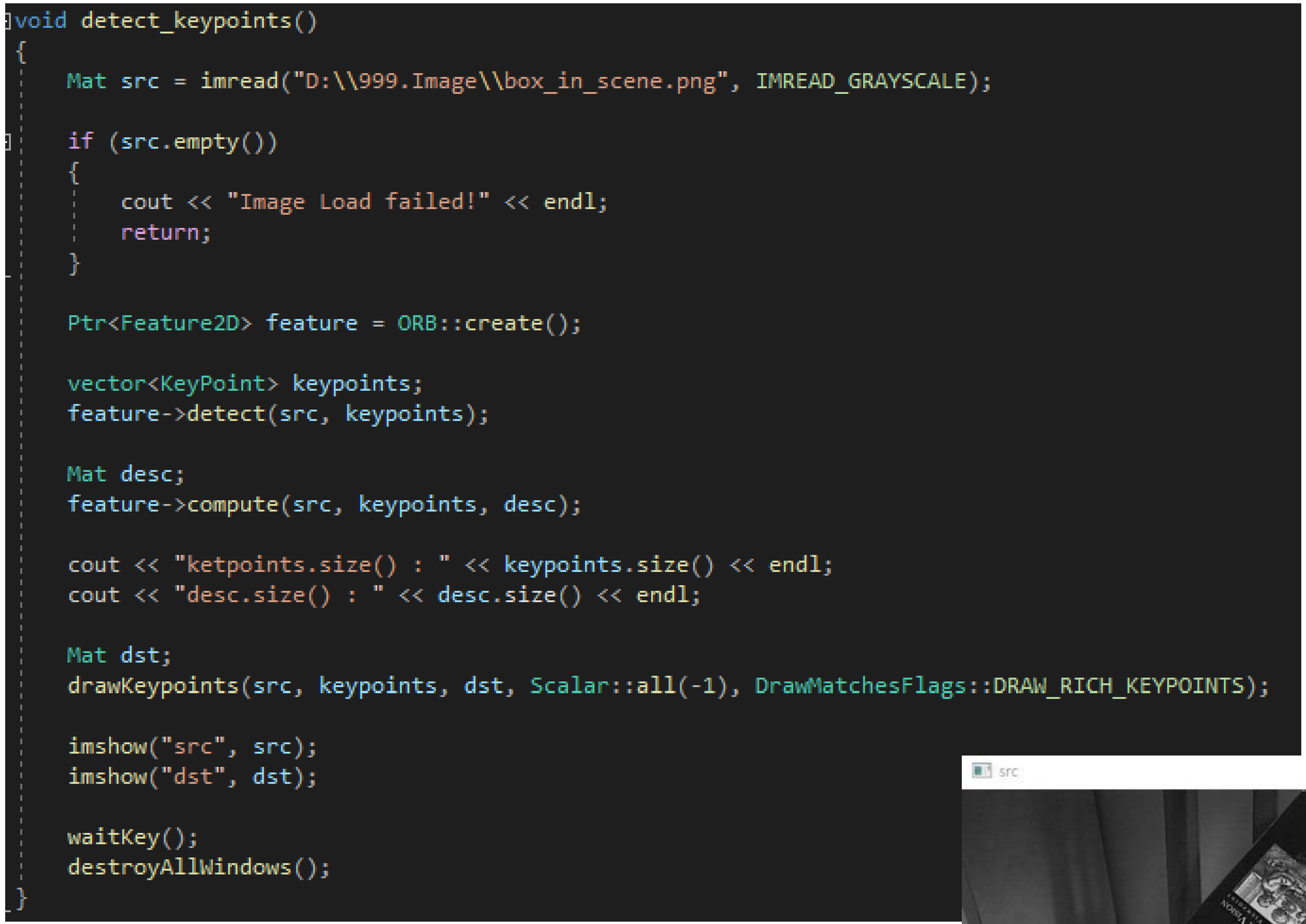
: 17.특징추출(2) – p.19



```
void corner_fast()
{
    Mat src = imread("D:\\999.Image\\building.jpg", IMREAD_GRAYSCALE);

    if (src.empty())
    {
        cout << "Image Load failed!" << endl;
        return;
    }

    vector<KeyPoint> keypoints;
    FAST(src, keypoints, 60, true);

    Mat dst;
    cvtColor(src, dst, COLOR_GRAY2BGR);
    for (KeyPoint kp : keypoints)
    {
        Point pt(cvRound(kp.pt.x), cvRound(kp.pt.y));
        circle(dst, pt, 5, Scalar(0, 0, 255), 2);
    }
    imshow("src", src);
    imshow("dst", dst);

    waitKey();
}
```



: 17.특징추출(2) – p.37



```
void detect_keypoints()
{
    Mat src = imread("D:\\999.Image\\box_in_scene.png", IMREAD_GRAYSCALE);

    if (src.empty())
    {
        cout << "Image Load failed!" << endl;
        return;
    }

    Ptr<Feature2D> feature = ORB::create();

    vector<KeyPoint> keypoints;
    feature->detect(src, keypoints);

    Mat desc;
    feature->compute(src, keypoints, desc);

    cout << "ketpoints.size() : " << keypoints.size() << endl;
    cout << "desc.size() : " << desc.size() << endl;

    Mat dst;
    drawKeypoints(src, keypoints, dst, Scalar::all(-1), DrawMatchesFlags::DRAW_RICH_KEYPOINTS);

    imshow("src", src);
    imshow("dst", dst);

    waitKey();
    destroyAllWindows();
}
```