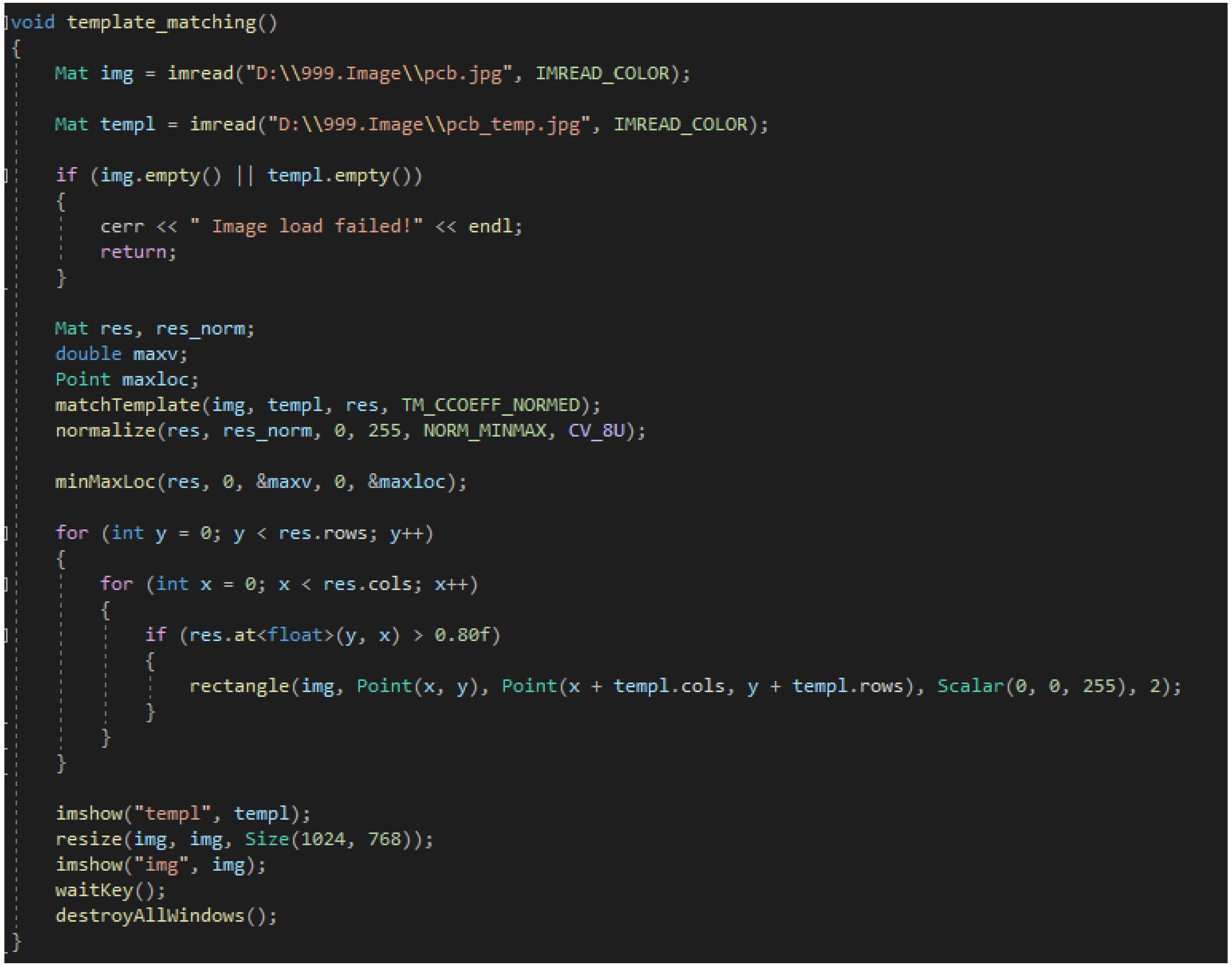
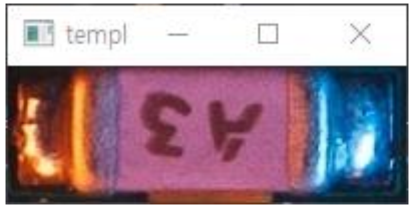
영상처리 실제 - 13주차 과제

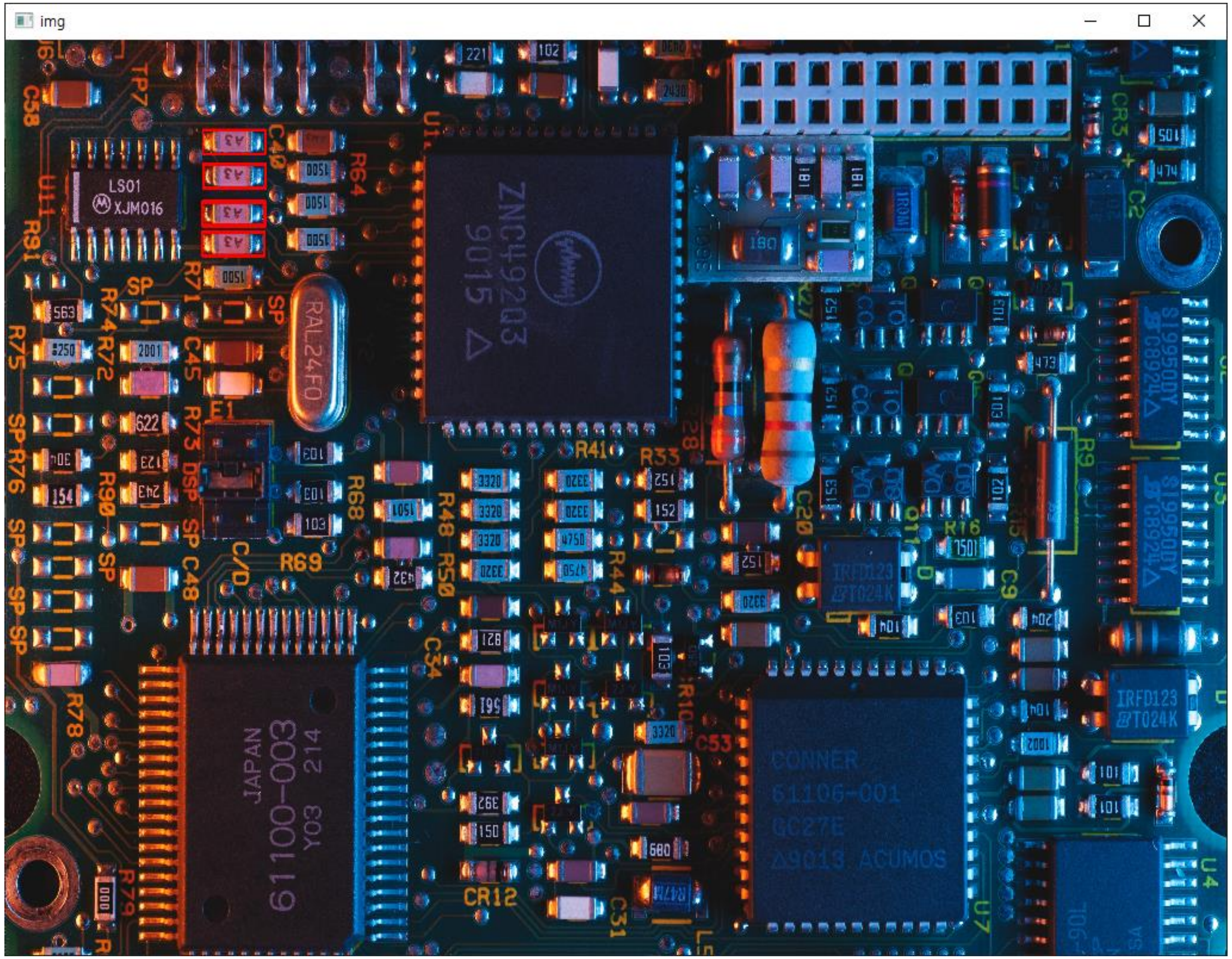
2023254009 최현동

: 18 – 특징매칭 – HW1

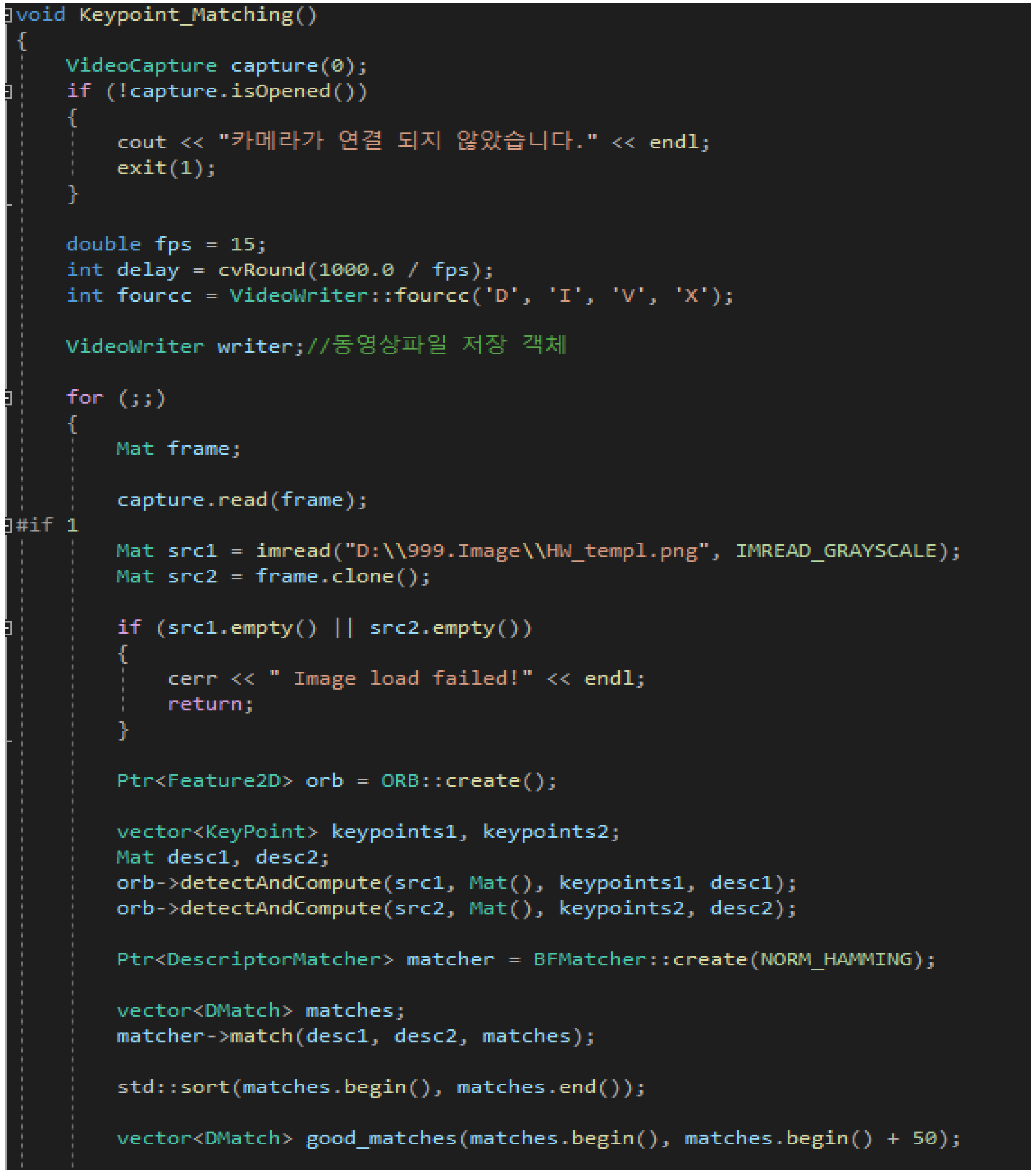

```
void template_matching()
{
    Mat img = imread("D:\\999.Image\\pcb.jpg", IMREAD_COLOR);

    Mat templ = imread("D:\\999.Image\\pcb_temp.jpg", IMREAD_COLOR);

    if (img.empty() || templ.empty())
    {
        cerr << " Image load failed!" << endl;
        return;
    }

    Mat res, res_norm;
    double maxv;
    Point maxloc;
    matchTemplate(img, templ, res, TM_CCOEFF_NORMED);
    normalize(res, res_norm, 0, 255, NORM_MINMAX, CV_8U);

    minMaxLoc(res, 0, &maxv, 0, &maxloc);

    for (int y = 0; y < res.rows; y++)
    {
        for (int x = 0; x < res.cols; x++)
        {
            if (res.at<float>(y, x) > 0.80f)
            {
                rectangle(img, Point(x, y), Point(x + templ.cols, y + templ.rows), Scalar(0, 0, 255), 2);
            }
        }
    }

    imshow("templ", templ);
    resize(img, img, Size(1024, 768));
    imshow("img", img);
    waitKey();
    destroyAllWindows();
}
```

: 18 – 특징매칭 – HW1

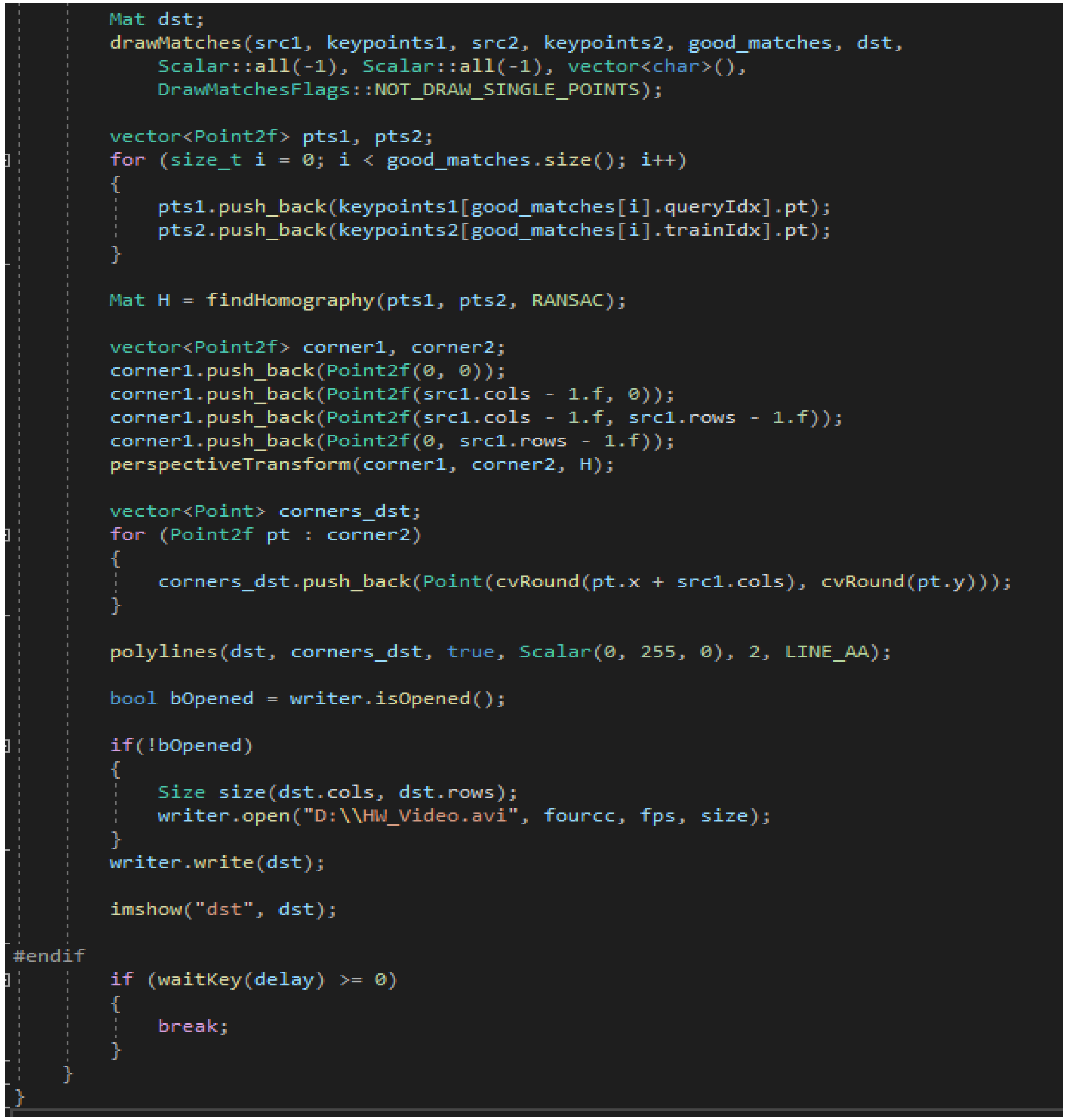




: 18 – 특징매칭 – HW2



```
void Keypoint_Matching()
{
    VideoCapture capture(0);
    if (!capture.isOpened())
    {
        cout << "카메라가 연결 되지 않았습니다." << endl;
        exit(1);
    }

    double fps = 15;
    int delay = cvRound(1000.0 / fps);
    int fourcc = VideoWriter::fourcc('D', 'I', 'V', 'X');

    VideoWriter writer;//동영상파일 저장 객체

    for (;;)
    {
        Mat frame;

        capture.read(frame);
#if 1
        Mat src1 = imread("D:\\999.Image\\HW_templ.png", IMREAD_GRAYSCALE);
        Mat src2 = frame.clone();

        if (src1.empty() || src2.empty())
        {
            cerr << " Image load failed!" << endl;
            return;
        }

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

        vector<KeyPoint> keypoints1, keypoints2;
        Mat desc1, desc2;
        orb->detectAndCompute(src1, Mat(), keypoints1, desc1);
        orb->detectAndCompute(src2, Mat(), keypoints2, desc2);

        Ptr<DescriptorMatcher> matcher = BFMatcher::create(NORM_HAMMING);

        vector<DMatch> matches;
        matcher->match(desc1, desc2, matches);

        std::sort(matches.begin(), matches.end());

        vector<DMatch> good_matches(matches.begin(), matches.begin() + 50);
```



```
        Mat dst;
        drawMatches(src1, keypoints1, src2, keypoints2, good_matches, dst,
            Scalar::all(-1), Scalar::all(-1), vector<char>(),
            DrawMatchesFlags::NOT_DRAW_SINGLE_POINTS);

        vector<Point2f> pts1, pts2;
        for (size_t i = 0; i < good_matches.size(); i++)
        {
            pts1.push_back(keypoints1[good_matches[i].queryIdx].pt);
            pts2.push_back(keypoints2[good_matches[i].trainIdx].pt);
        }

        Mat H = findHomography(pts1, pts2, RANSAC);

        vector<Point2f> corner1, corner2;
        corner1.push_back(Point2f(0, 0));
        corner1.push_back(Point2f(src1.cols - 1.f, 0));
        corner1.push_back(Point2f(src1.cols - 1.f, src1.rows - 1.f));
        corner1.push_back(Point2f(0, src1.rows - 1.f));
        perspectiveTransform(corner1, corner2, H);

        vector<Point> corners_dst;
        for (Point2f pt : corner2)
        {
            corners_dst.push_back(Point(cvRound(pt.x + src1.cols), cvRound(pt.y)));
        }

        polylines(dst, corners_dst, true, Scalar(0, 255, 0), 2, LINE_AA);

        bool bOpened = writer.isOpened();

        if(!bOpened)
        {
            Size size(dst.cols, dst.rows);
            writer.open("D:\\HW_Video.avi", fourcc, fps, size);
        }
        writer.write(dst);

        imshow("dst", dst);

#endif
        if (waitKey(delay) >= 0)
        {
            break;
        }
    }
}
```

: 18 – 특징매칭 – HW2