计算机视觉和模式识别 作业 3

13331231
孙圣
计应 2 班

一、使用说明

MAC OSX 系统：通过 sh execute.sh 直接编译运行即可（需要安装 opencv）,默认对第 1 张图片进行处理。要对其他图片进行测试，执行./a.out ./Dataset/X.jpg 即可。

Windows 系统，在 cmd 中输入 execute.exe ./Dataset/X.jpg。

二、实验过程

这次对 A4 纸的矫正大部分是基于上次的边缘和边缘点的检测。因此上次作业的正确性会大大影响这次作业的成功率。

一开始的时候，我就发现了上次作业中的几个 Bug。对于竖直的直线来说，theta 可能的值可能接近 0，也可能接近 PI，这就对直线去重那段程序带来了很多麻烦。由于我是先将所有直线大致按照 theta 进行排序，当 theta 比较接近时，则按照 rho 进行排序。这样，theta 为 0 的线段会被排在前面，而 theta 接近 PI 的线段会排到后面。因此我就在去重之前，先将 theta 接近 PI 的线段移动到前面。但这里需要注意的是，在把直线移到前面的时候，要考虑 rho 的大小。一开始没有考虑的这个问题，所以 Dataset2/10.jpg 一直不能成功去重。

还有一个问题就是，对于缺了角的 A4 纸，很容易在 Hough 变换中会把缺角的边检测到。因此要进行错误线段的排查。这里我用的方法是：如果一条线段与他前面和后面的线段的 theta 值相差都过大时，则认为这条线段是错误检测的线段，因此要删去。

将上次作业中的 Bug 改正后，才进入到这次作业的编程之中。

首先，我的目标是将所有的 A4 纸都旋转到正的方向。因此考虑使用排序后的第一条线段的 theta 值所对应的角度进行旋转。但这样会产生一个问题，对于 Dataset 中的第二张和第四张图片，他们旋转后都会呈水平方向，因此还要对他们顺时针旋转 90 度。而对于那些本身就是横着放的图片，则也要顺时针旋转 90 度，但这样做的话，有些图片的会超出原来的边界，因此我考虑将它们缩小一定的倍数。以上的旋转都是调用 opencv 中的

```
rotationMatrix = getRotationMatrix2D(center, rotationAngle, 1);
warpAffine(srcImg, rotatedImg, rotationMatrix, rotatedImg.size() );
```

函数来实现的。其中还要注意的是旋转的中心是 A4 纸的中心，不然也很容易在旋转的过程中图片超出边界。对图像旋转完成后，也需要对四个坐标点进行相应的变换，这里利用了

```
transform(pts, pts, rotationMatrix);
```

之后，对转换后的点进行排序，这样可以得到一个稳定的顺序，即左上，右上，左下，右下。然后设定四个与之对应的标准点，然后调用

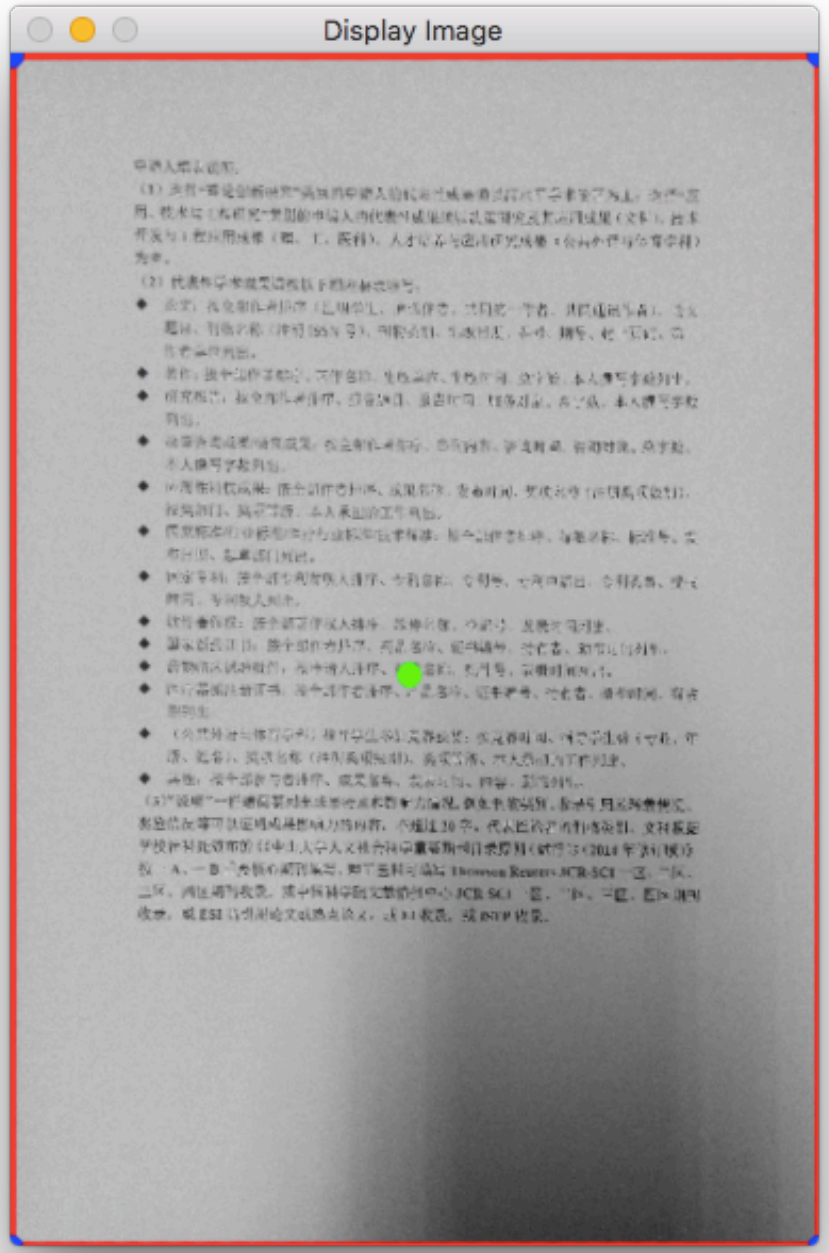
```
Mat perspectiveMatrix;
perspectiveMatrix = getPerspectiveTransform(pts, standardPoints);

Mat perspectiveImg(800, 800, rotatedImg.type());
warpPerspective(rotatedImg, perspectiveImg, perspectiveMatrix, perspectiveImg.size());
```

完成矫正。

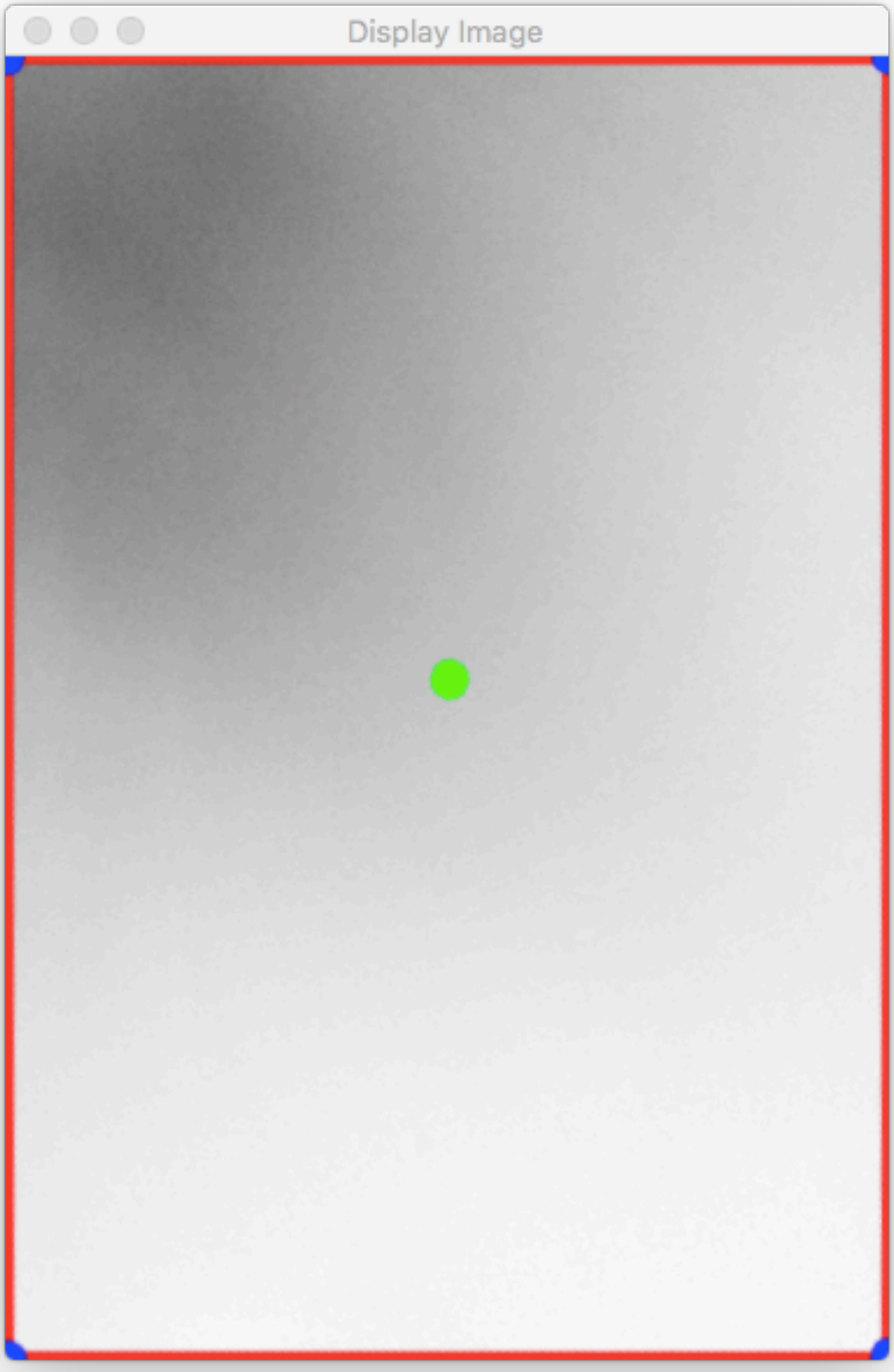
最后定义一个 Rect,将图片裁剪出来并显示。

三、实验结果

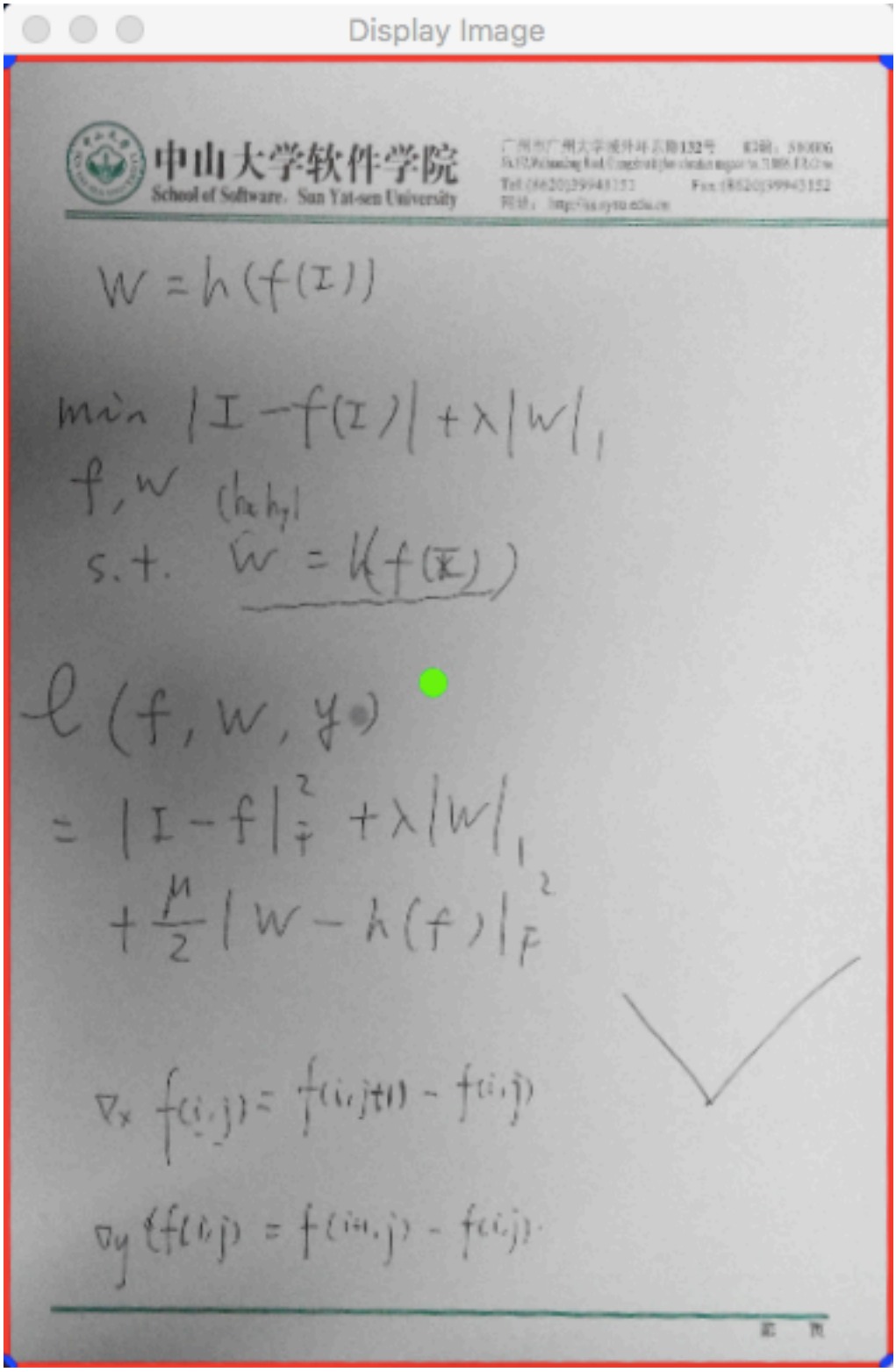
Dataset/1.jpg



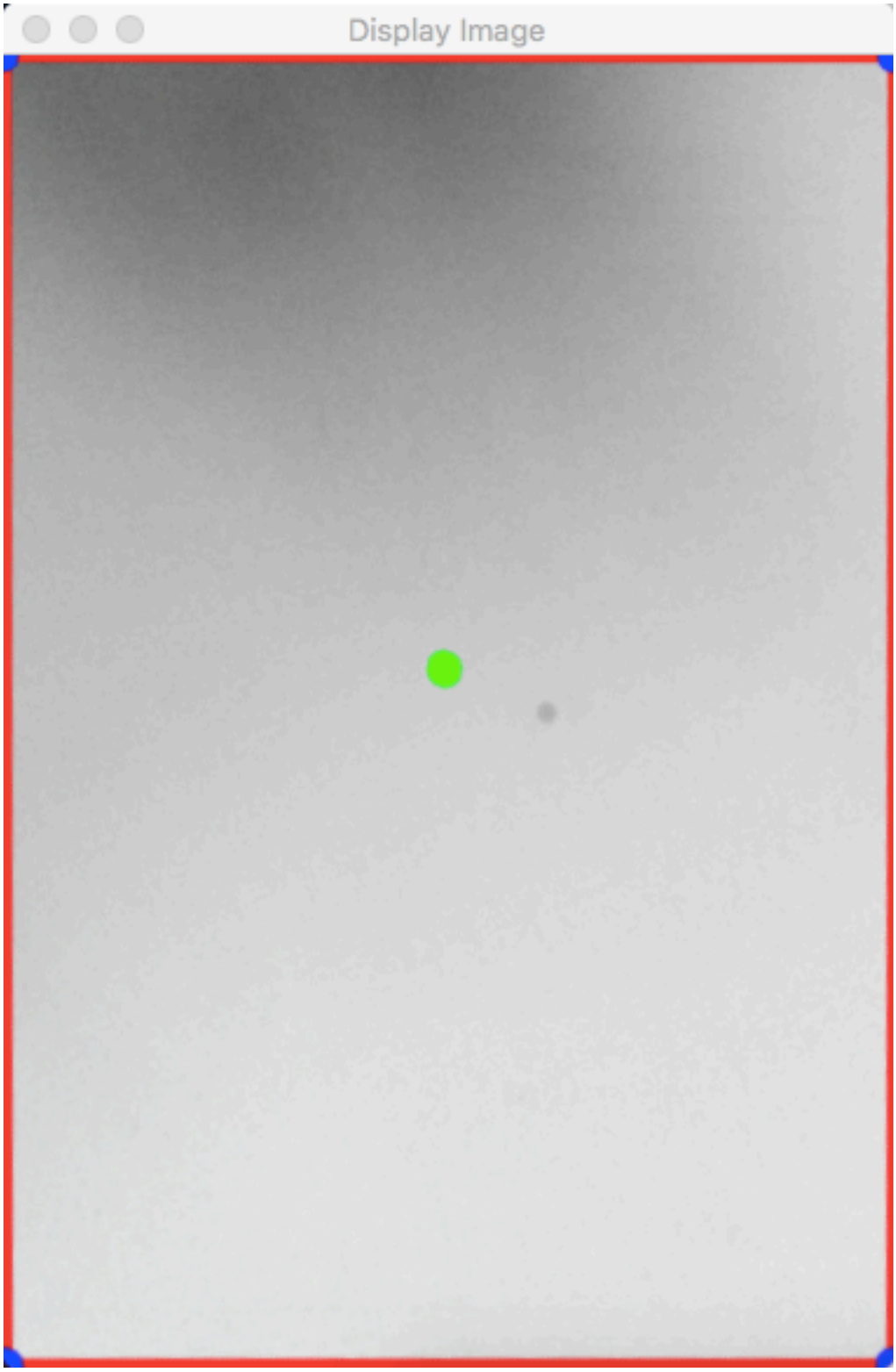
Dataset/2.jpg



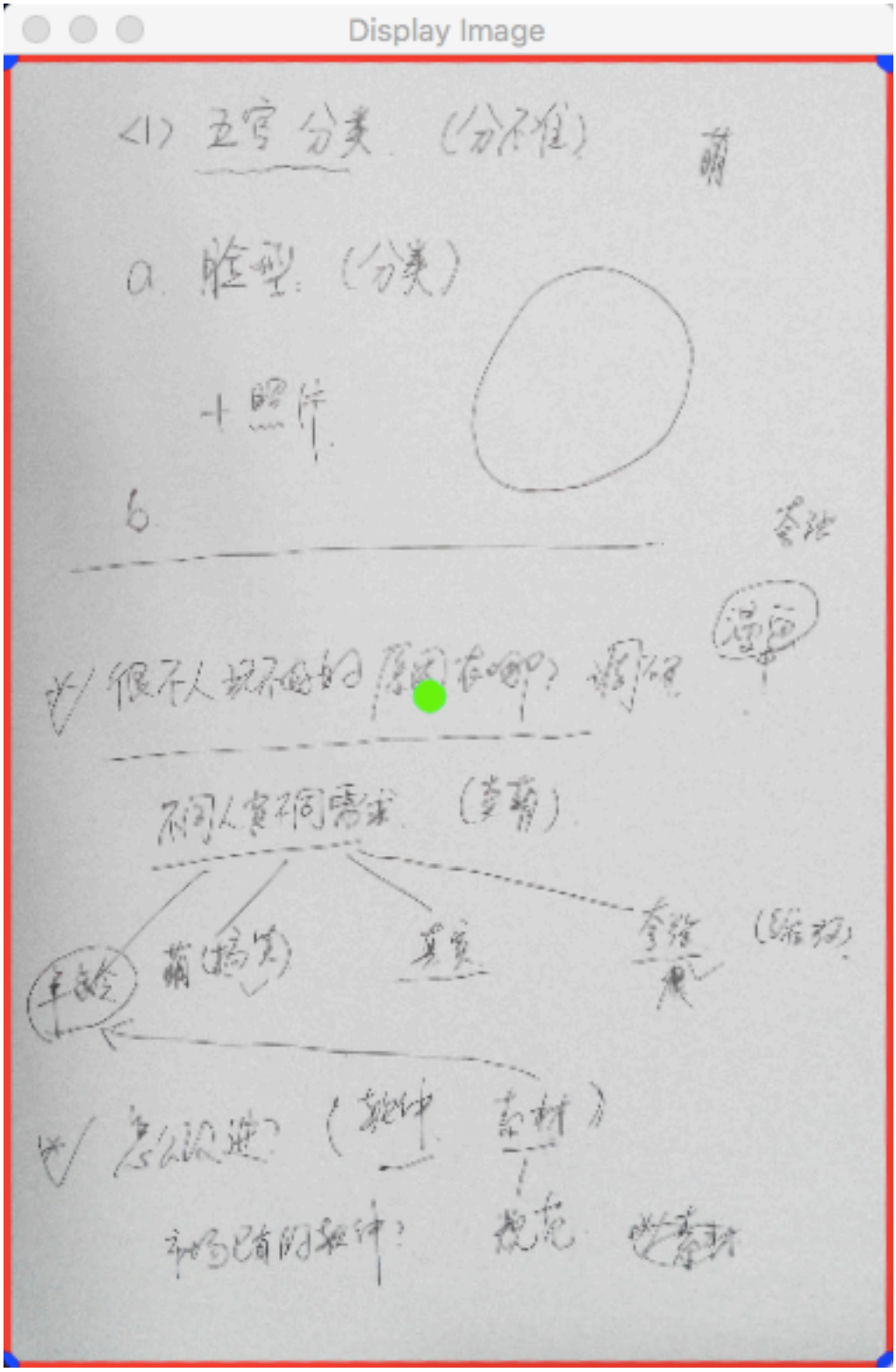
Dataset/3.jpg



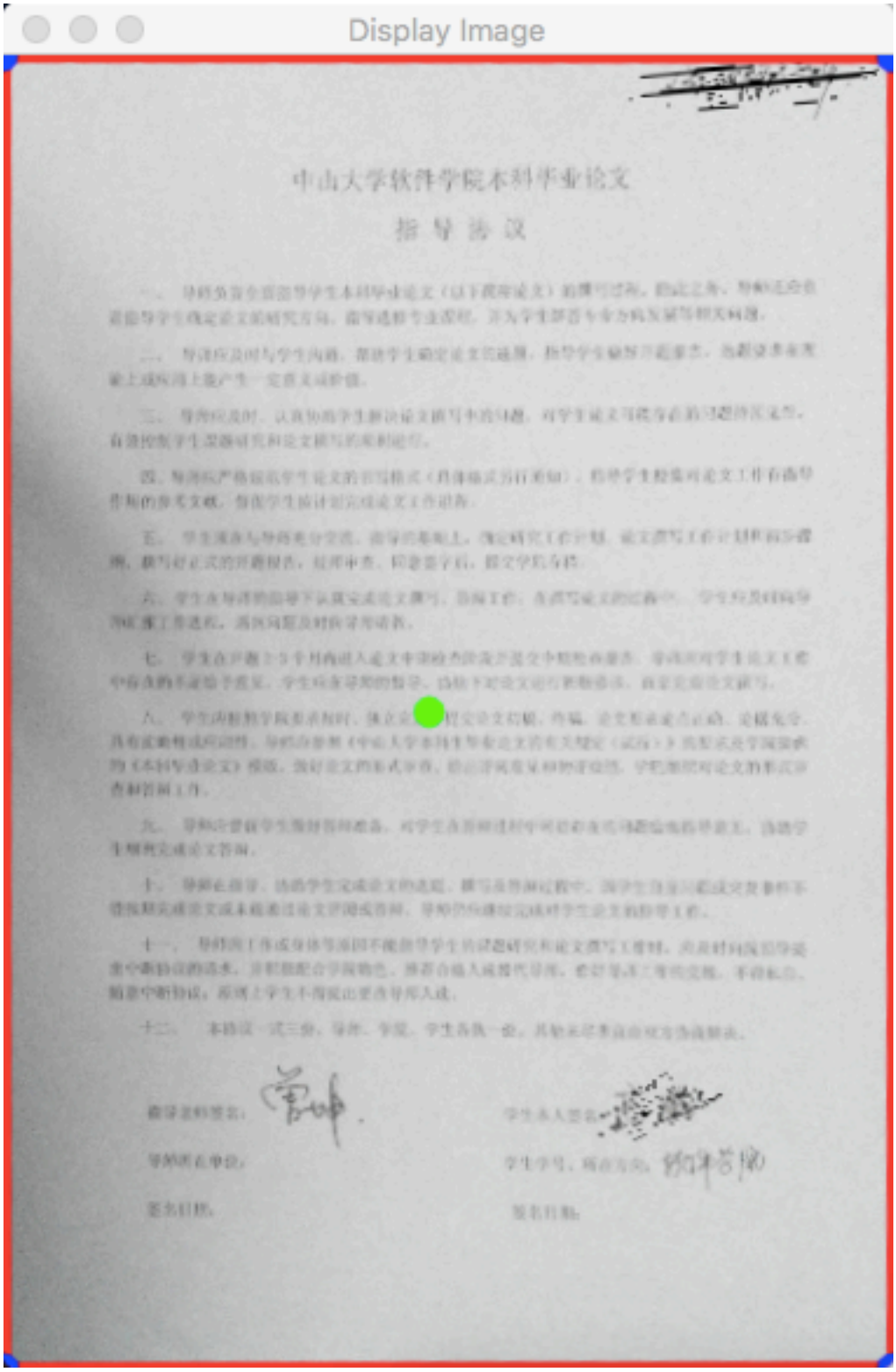
Dataset/4.jpg



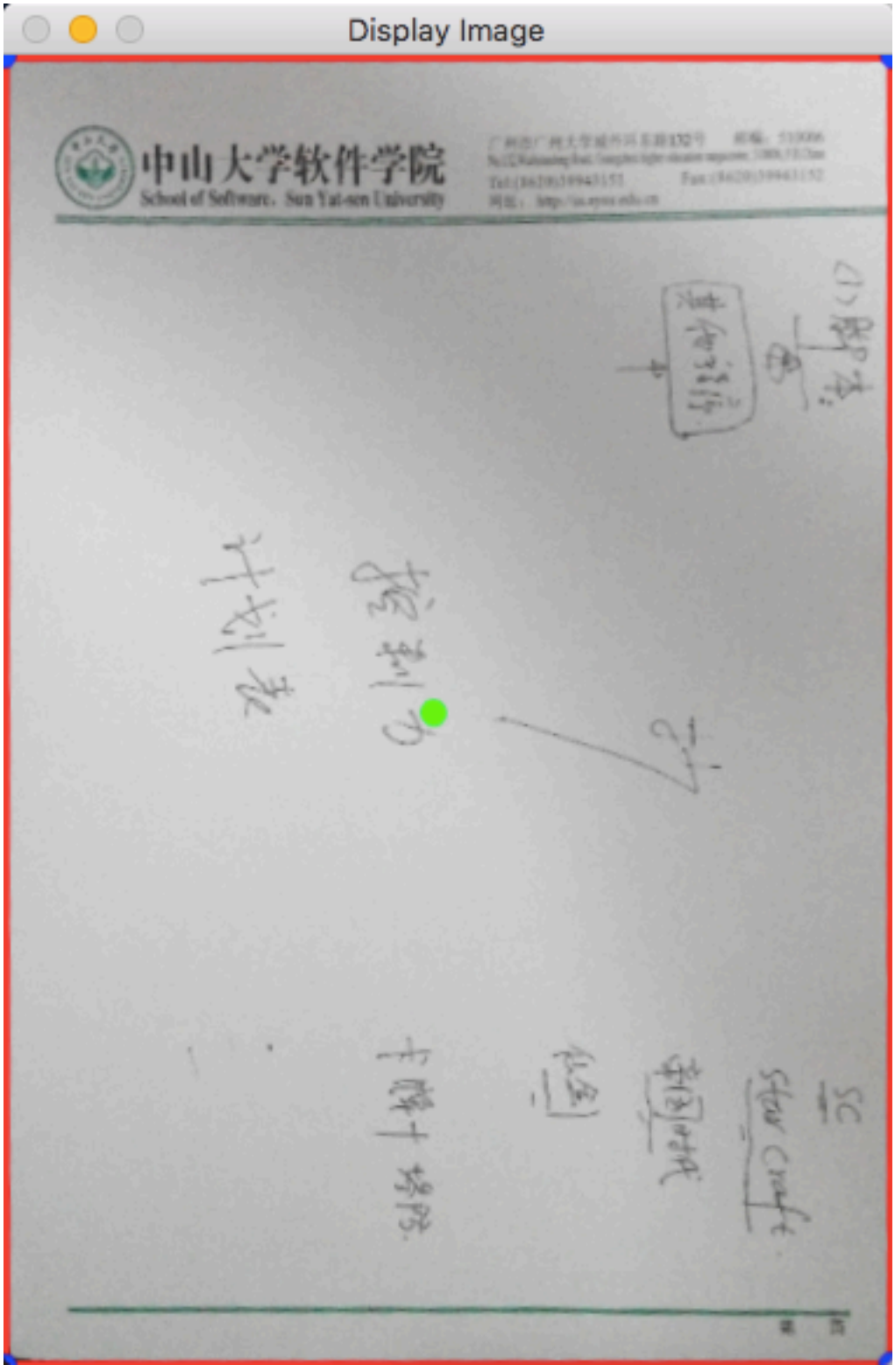
Dataset/5.jpg



Dataset/6.jpg

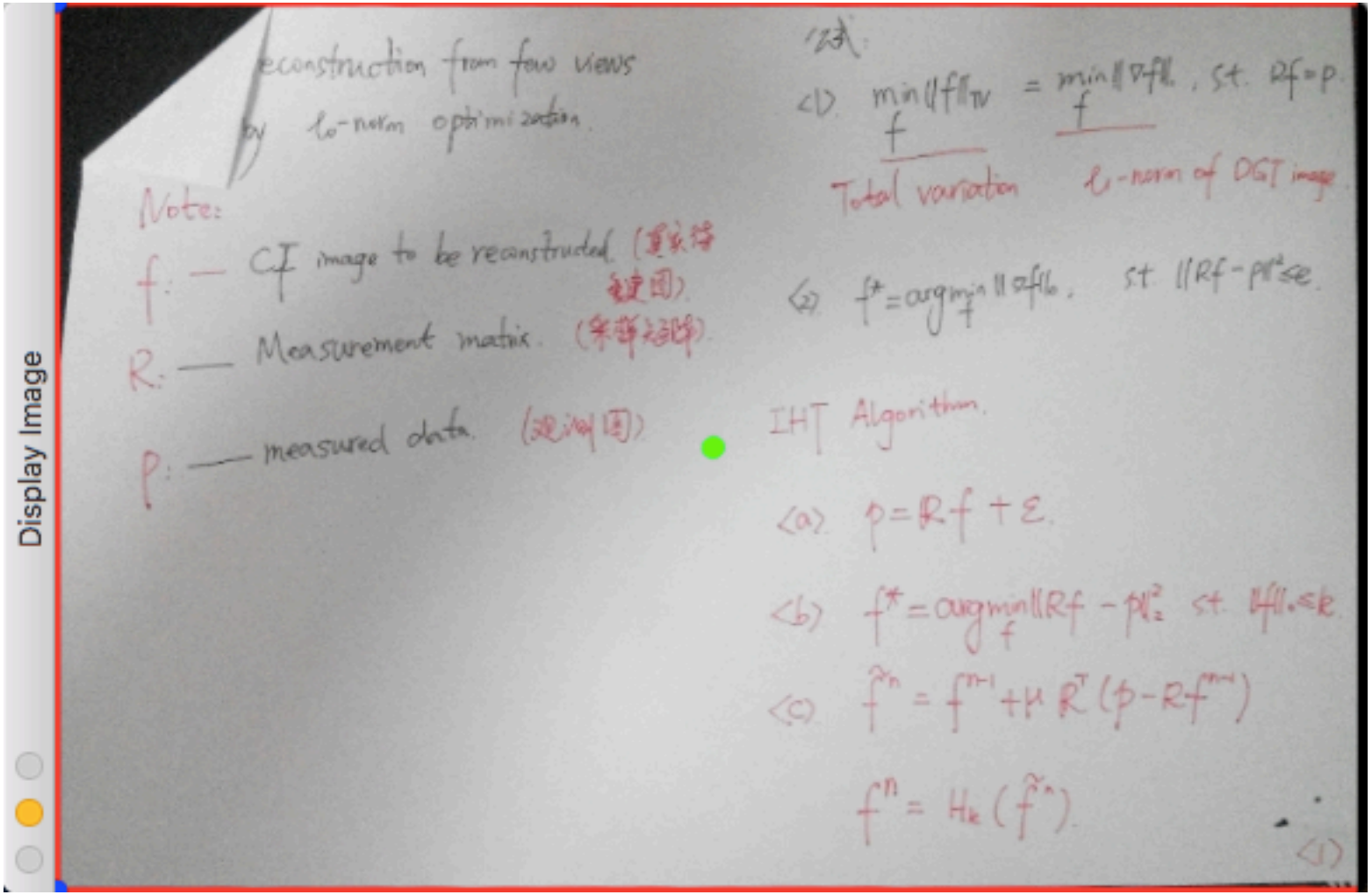


Dataset2/1.jpg

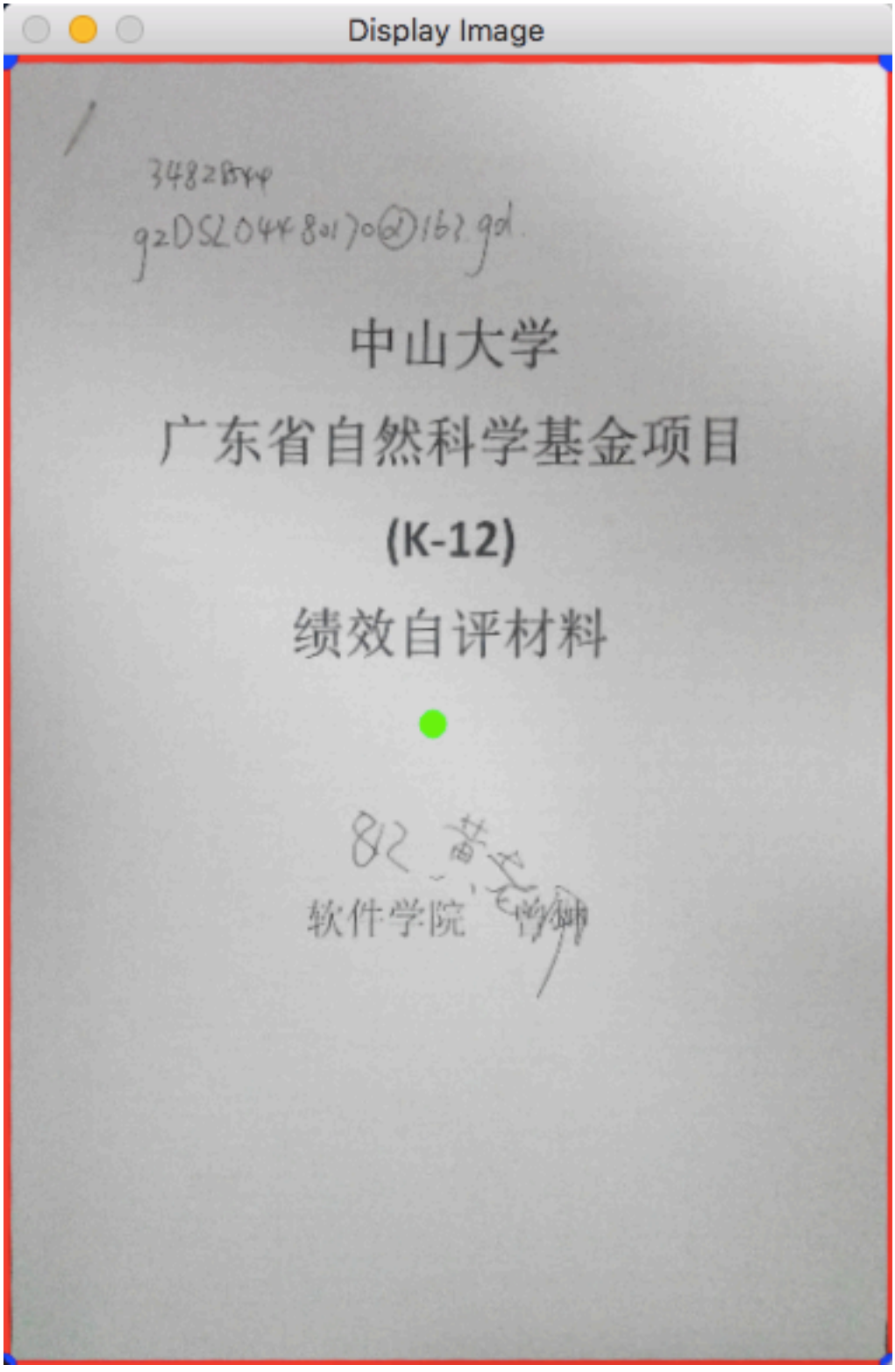


Dataset2/2. jpg

Display Image



Dataset2/3.jpg



Display Image

$d_{ij} = x_i^T \cdot x_i + x_j^T \cdot x_j - 2 x_i^T \cdot x_j$

$= b_{ii} + b_{jj} - 2 b_{ij}$

算法基于下面的假设：

<1> 聚类中心被一些低密度的点包围。

对每个数据，求两个指标：

<a> ρ_i：局部密度

<b> δ_i：从 i 到其高密度的距离

$\rho_i = \sum_j \chi(d_{ij} - d_c)$

距离小于 d_c 的点的数目

这里什么是 d_c？怎么计算？

d_c：cutoff Distance.

截断距离

d_c 一定要是全局的么？

可以为 Adaptive 的么？

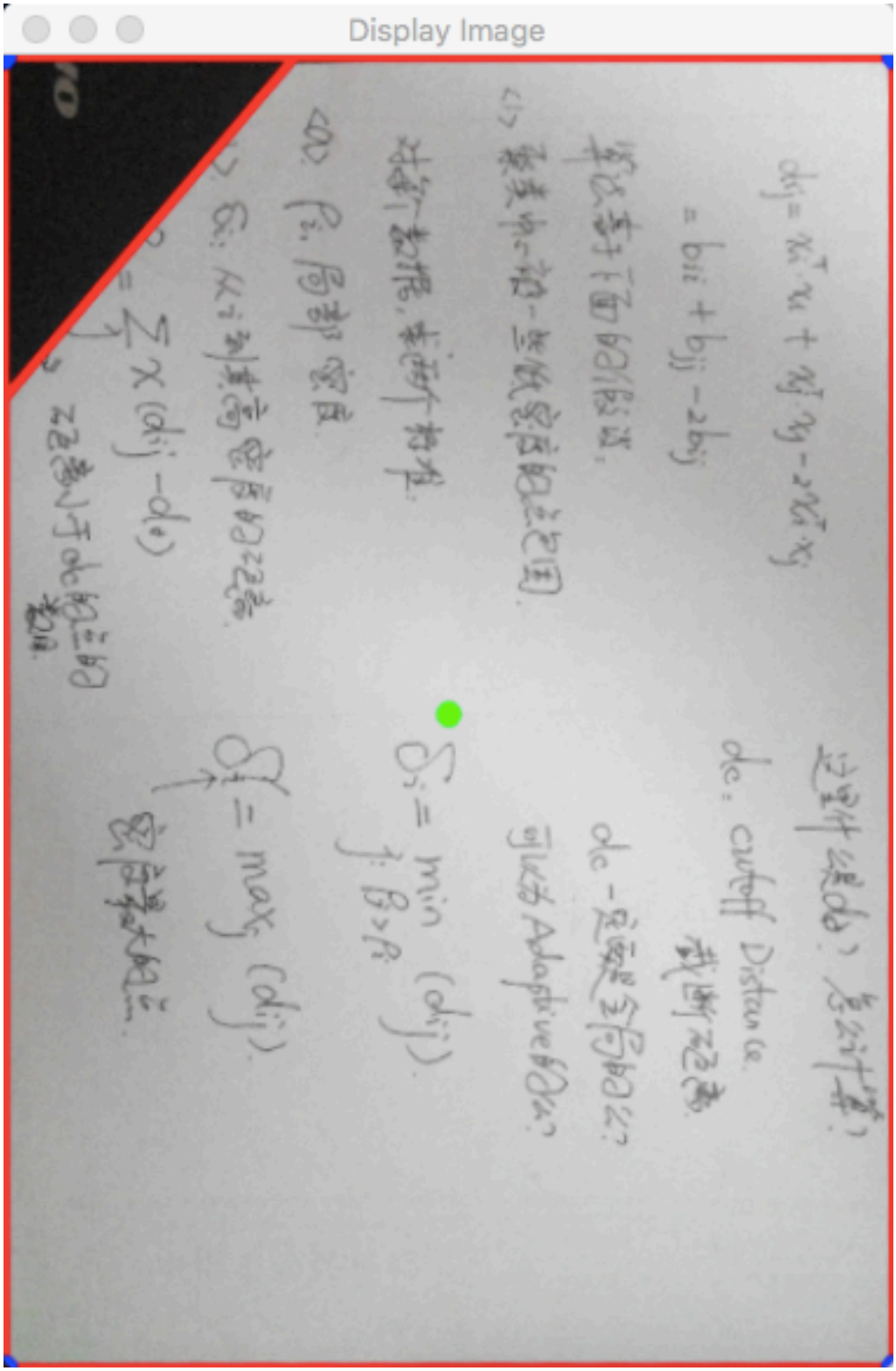
$\delta_i = \min_{j:\rho_j > \rho_i} (d_{ij})$

$\delta_i = \max_j (d_{ij})$

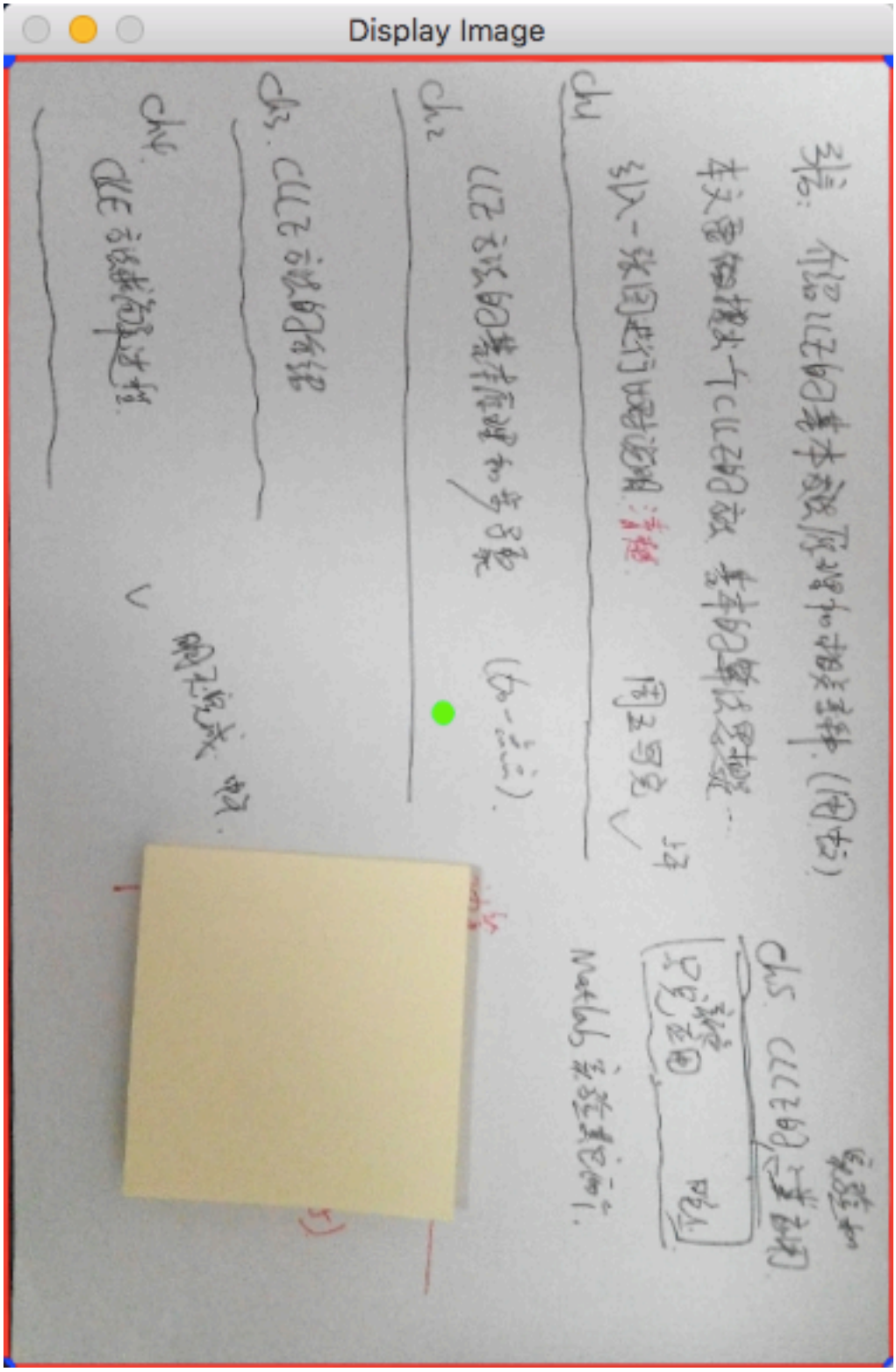
密度最大的点

Dataset2/4.jpg

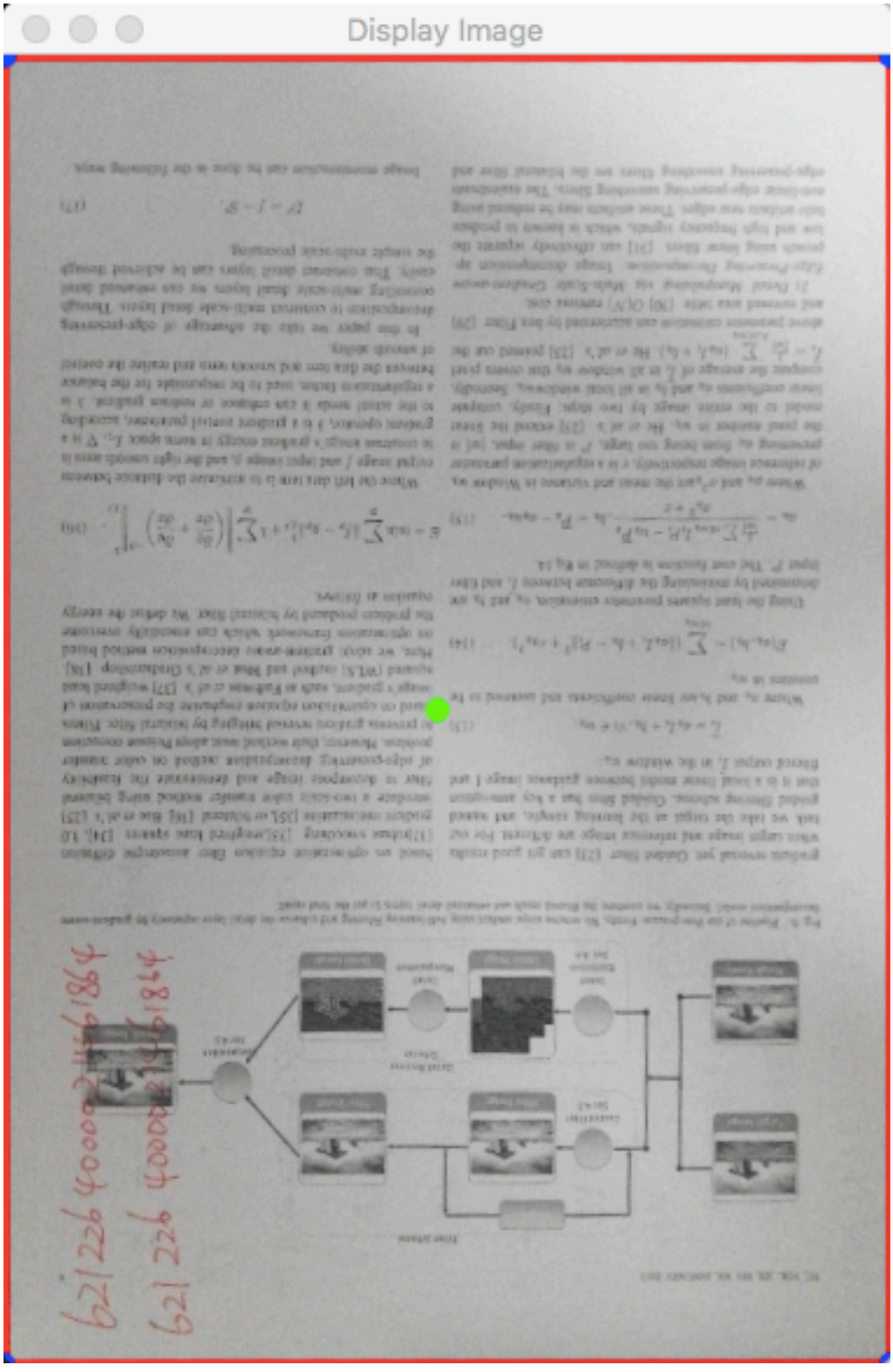
Dataset2/5.jpg



Dataset2/6.jpg

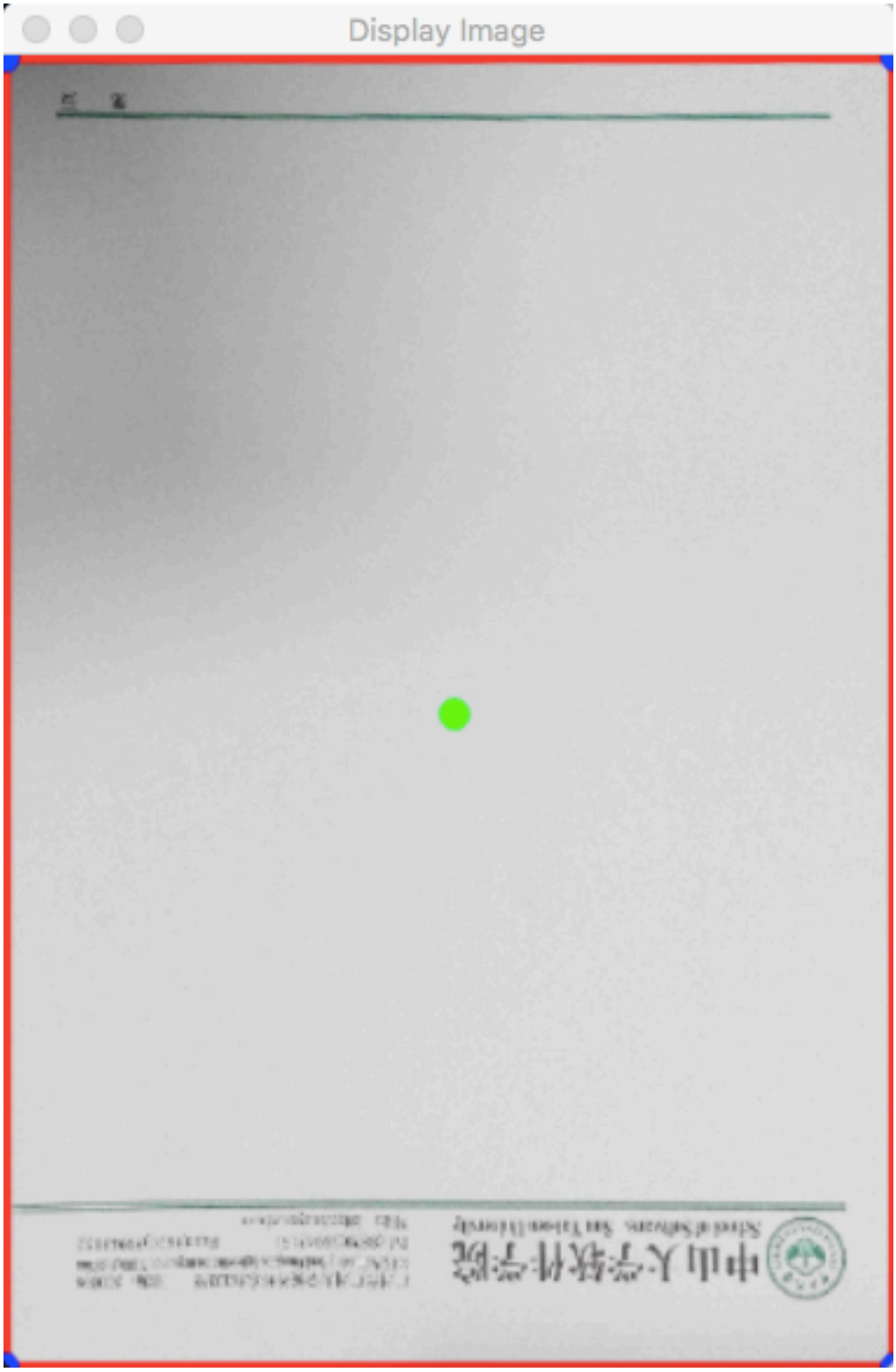


Dataset2/7.jpg

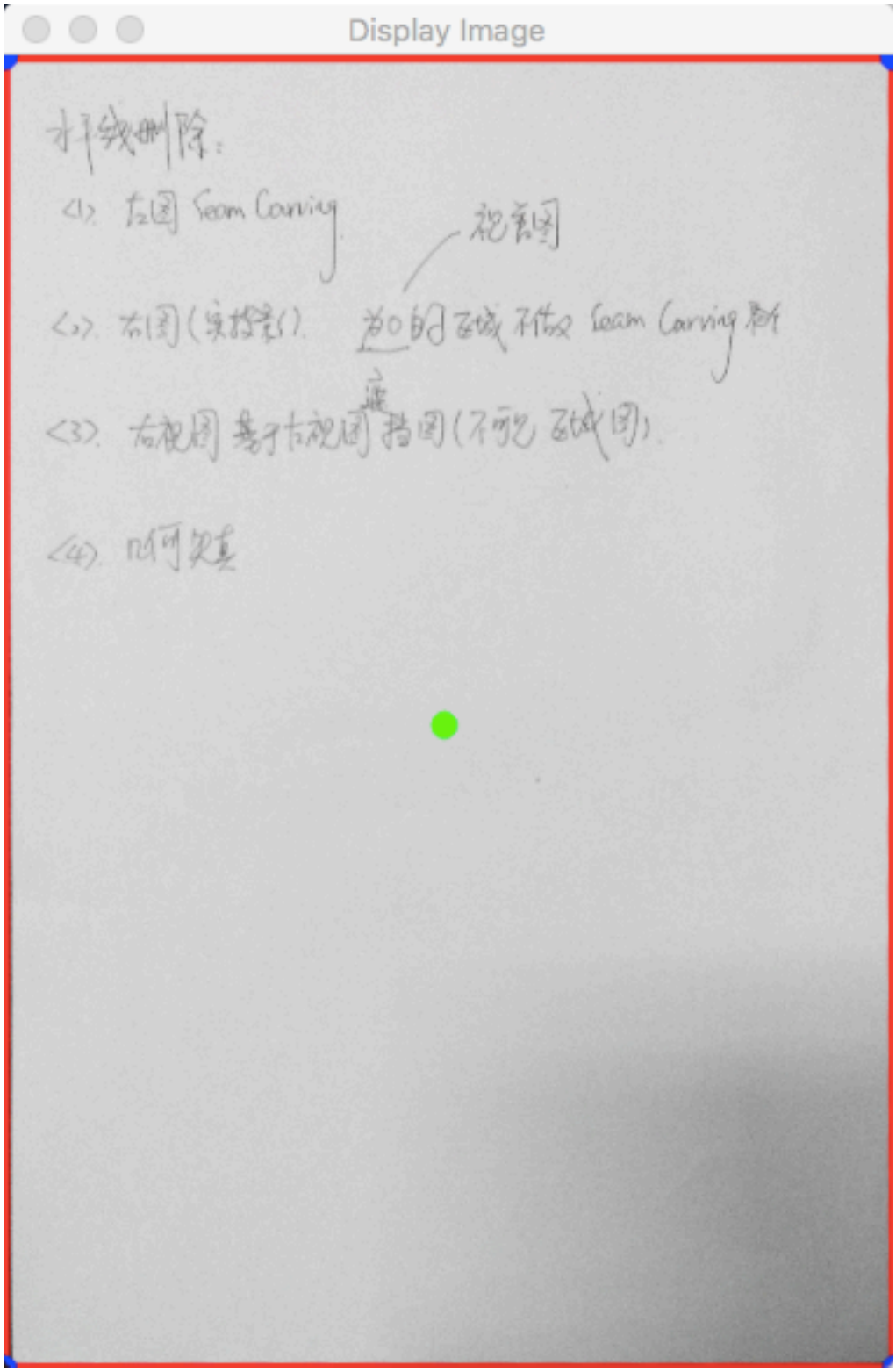


Dataset2/8.jpg

Dataset2/9.jpg



Dataset2/10.jpg



参考资料：

[1] warpAffine
http://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/warp_affine/warp_affine.html
[2] transform
http://docs.opencv.org/2.4/modules/imgproc/doc/geometric_transformations.html
[3] transform 函数中的 Assertion failed 解决办法
https://groups.google.com/forum/?fromgroups=#!topic/android-opencv/CPDMJsmYVBI
[4] 图像裁剪
http://stackoverflow.com/questions/8267191/how-to-crop-a-cvmat-in-opencv
[5] perspective transform 原理
http://math.stackexchange.com/questions/296794/finding-the-transform-matrix-from-4-projected-points-with-javascript