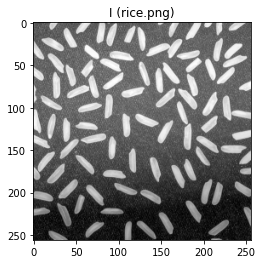
**《图像分析与理解》课程——图像配准相关实验**

212138 卓旭

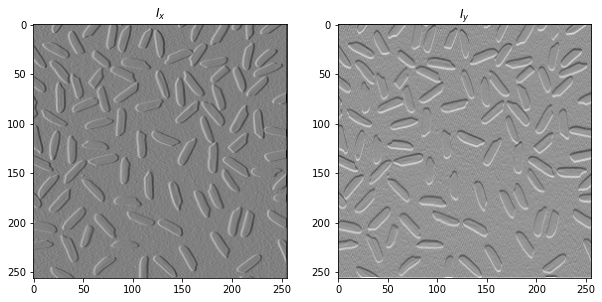
**Part 1 – Harris角点检测**

1. 读取rice.png，结果如下：

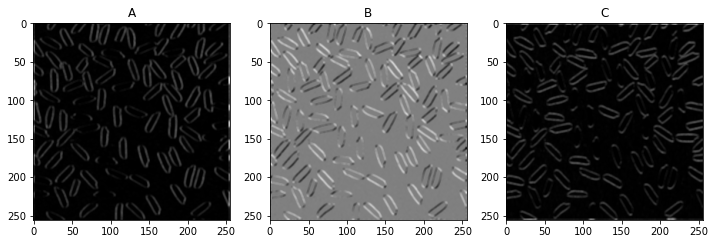


2. 计算每个像素的梯度形成梯度图。按照梯度定义

可以设计卷积核分别计算*x*和*y*方向的梯度图（实际上我们可以选用长条形的1×3、3×1卷积，但方形的3×3卷积更容易理解）。将转为float32数据类型后，使用cv2.filter2D函数完成卷积，设置borderType参数为BORDER\_CONSTANT以对图像外圈补零，保证所有原始像素点均可求得梯度。结果如下：



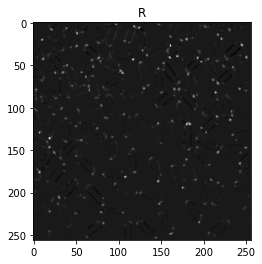
3. 计算邻域内的梯度二次项。首先求出，然后构造卷积核分别卷积三张二次项图来实现3×3邻域求和。进一步，使用cv2.filter2D函数完成卷积，同样对图像外圈补零，得到、、结果如下：



4. 用*A*、*B*、*C*的值的构成分块矩阵*M*

则可求取角点响应（corner-ness）*R*值

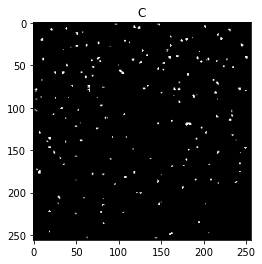
代入各图，结果如下：



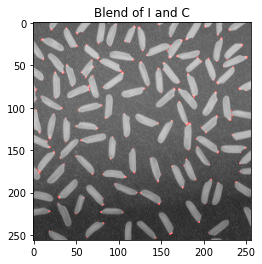
由于求取值的参数是在3×3邻域操作后得到的，因此我们这里忽略了图像四周2像素宽的一圈（第一步求梯度图时引入1像素padding，第三步邻域求和时引入1像素padding，综合导致这里2像素宽的四周值无效），将其置0，因为那些数据不准确。

另外要注意，这里使用的是邻域求和后的图像参与运算，而不是求Hessian矩阵本身。

5. 统计得到。求得，则阈值。将大于该阈值的点认为是角点，得到角点图如下：

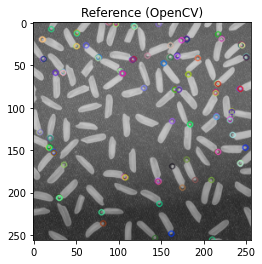


6. 将灰度图*I*转为RGB图，然后将*C*的幅度增强后叠加到R通道，即得到角点被红色标注出的结果图，如下：



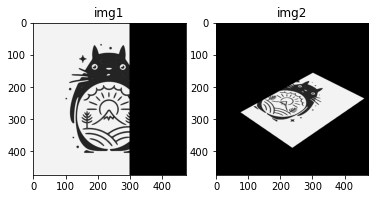
可以看到该算法确实检测出了图像中存在的“角点”位置。

后面我使用OpenCV的Harris角点检测器，使用相近的参数得到参考图像如下，两相对比，说明我们的结果正确。



**Part 2 – 透视变换**

1. 读取img1和img2，利用之前实验的子程序即可完成，结果如下

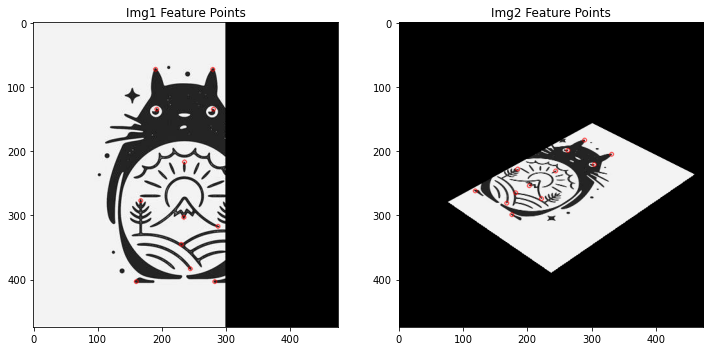


2. 我们试图确认一组img1到img2的变换，使得两图可以拼合出完整的原图。

3、4. 手工确认一系列img1与img2之间应该匹配的特征点，共12个，列表如下（标☆的是推荐点）：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **序号** | ***img1 (x1,i, y1,i)*** | | ***img2 (x2,i, y2,i)*** | |
| ***x* (row)** | ***y* (col)** | ***x* (row)** | ***y* (col)** |
| 1☆ | 73 | 190 | 183 | 288 |
| 2☆ | 73 | 279 | 205 | 330 |
| 3☆ | 135 | 192 | 199 | 262 |
| 4☆ | 135 | 280 | 221 | 303 |
| 5☆ | 277 | 167 | 228 | 184 |
| 6☆ | 317 | 287 | 274 | 221 |
| 7☆ | 403 | 160 | 262 | 118 |
| 8☆ | 403 | 282 | 299 | 175 |
| 9 | 217 | 235 | 231 | 243 |
| 10 | 303 | 234 | 254 | 202 |
| 11 | 383 | 244 | 281 | 167 |
| 12 | 345 | 230 | 265 | 181 |

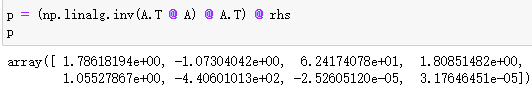
将各点在图上用红圈标出后如下：



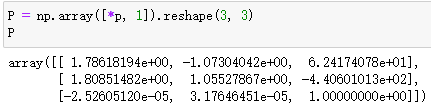
5. 接下来，目的是寻找变换

，其中

按照作业讲义的说明，排布矩阵*A*、向量*p*以及右手边，*A*有2×12行8列：，右手边是长为24的向量，然后得到方程组，进而利用*A*的伪逆求解之，得到*p*：



对应的矩阵*P*为：



6、7. 我们遍历假设已知的变换后目标图像中的所有坐标的齐次形式，在该向量左乘*P*，即得到向量，伸缩将第三元化为1（）即可求得对应原图像上的坐标，对此像素采样即得到在该处的值。

为了完成这一步，对在亚像素上的点作快速最近邻插值（*x*、*y*坐标直接四舍五入），超出原图像范围的像素点置0，反复计算可得如下：



可见该步骤完成了img1向img2的几何的透视变换过程。

8. 我们认为足够准确，此时只需要与img2对各像素点对点计算最大值，即可得到融合后的图像img3，结果如下：



整套流程基本完成了img1与img2的全图拼合过程，但仍存在一些误差，应该可以通过更多的特征点选择来优化。