票据边框检测算法检讨

我们目前的问题在于,霍夫变换函数得出的线段数量庞大,直接进行交点计算和矩形检测的计算量不堪重负,因此我们需要解决的核心问题是，如何减少线段数量**?**

因此,我们在之前着力于如何减少线段数量的手法上,如:

1. 邻近线段分类. 因需要不断迭代比较且对不同图例的效果不安定,故而暂时放弃.
2. 端点聚类求线段分类. 目前虽能得到较好的线段分类效果,但线合并后会出现偏移,结果不甚理想.

到此,我开始意识到,线段不管分类有多精准,线合并算法/规则都会成为我们下一道坎.

一个又一个问题接踵而至,迫使我不得不重新审视我们面临的问题:"如何减少线段数量?"

在我重新翻看dropbox提供的技术文档(地址: https://blogs.dropbox.com/tech/2016/08/fast-and-accurate-document-detection-for-scanning/)后,我发现他们的文档中并没有记述有关霍夫变换得出的线段过多,需要合并的问题. 同时,从文中看来,霍夫直线检测的过程,不深究其理的话,更像是一个单纯的阈值比较过程. 那么,是否有可能,我们的问题不是在于线,而是在于求线的过程呢?

查看过骚蓝提供的代码后,我了解到,霍夫直线检测的过程是由opencv提供的函数 `cv2.HoughLinesP()` 完成的. 关于该函数的参数说明见: https://blog.csdn.net/dcrmg/article/details/78880046 ;应用实例见: https://blog.csdn.net/on2way/article/details/47028969 .

在大致了解该部分的上下游之后,我有以下几个疑问:

1. 为什么使用逆二值化? 在多数实例中, 大都是用二值化处理
2. 为什么没有使用canny等方法做边缘检测? 一般的处理流程是使用 `cv2.Canny()` 函数进行边缘检测后的结果作为霍夫直线检测的输入 (扩展) 是否存在比canny更好的检测方法? (如dropbox文档中所写的基于ml的检测方法)
3. cv2.HoughLinesP的参数(rho,threshold等)是否合理? 参数说明写道, threshold, 累加平面阈值参数, 其值越大, 检出线段越长, 检出线段个数越少

以上,就是我目前为止对边框检测算法的的检讨.