一、初衷

刚开始有这个想法是18年五月份，当时微信跳一跳的辅助很火，自己也玩了几把，结果被同学一阵虐，发誓做个辅助找回尊严，刚好实验室的一位学长在空间分享他刚完成的跳一跳辅助的视频，于是就开始搜集资料准备开干，但是我这人一般别人做过的我就懒得做了，正好班里有人在玩腾讯桌球，感觉更有挑战性，网上也比较少见，于是便开始了。

二、构想

准备了开始首先就是搜集资料，借鉴一下微信跳一跳的方案。发现大多都是从通过ADB驱动，手机截屏，传到PC进行处理，而后再ADB模拟触控。其间的难点在于小人的识别与下一个建筑顶面的识别，而后就可通过标定（实验测出点按时长与跳跃距离的关系），实现自动运行。

我首先观察了一下桌球原本的一些辅助，可以看到已经具备了预撞击的位置，被撞球与白球的预测方向，但是太短了（很多同学拿着尺子做延长线）。经过一番分析和研究，我决定把两条预测线延长，这个就是初步的想法，当然如果顺利的话再做适当的提升。



三、思路

要实现上面的构想首先要解决的是如何获取到游戏的图像，之后在图像中找到所有球的位置（这里以圆代替），之后根据这些圆的特征分辨出不同的球（白球、瞄准环、被撞的球以及其他球），最后根据弹性碰撞原理，被撞的球会沿着两球质心所在直线运动，作出延长线。

**如何获取游戏图像？**方法有多种，下面都是我尝试过的方法。

1.ADB驱动截屏，再传输到PC。这种方式的问题是速度太慢，在跳一跳中可能还不明显，但桌球在瞄准时需要不断地调整角度，要求能够实时反馈调整的效果，这时就不够用了。（之所以要求实时性，在于我设计这个辅助是基于游戏自带的瞄准环的）

2.投屏，将手机屏幕通过投屏软件投射到PC，再由PC截图。这种方式相较于ADB更快，但是付出的代价是分辨率与画质降低，会占用很多网络（因为要实时传输视频），对PC和手机的性能影响都比较大。

3.模拟器，在PC端运行腾讯手游助手，安装微信，打开小程序，腾讯桌球，由截屏获取游戏图像。这样做是最合适的，摆脱了手机的限制，模拟器运行的游戏在PC端是实时的，截图可以获取实时图像。其次游戏的分辨率可调。

**如何在图像中找到所有的球？**

第一想法是找圆就可以了，但是仔细看图像中的圆不少，大小有些和球差不多，并不容易区分。那么这个圆就要加个限定了——在桌布上的圆。桌布的颜色人眼看来很容易区分，但是同时也注意到桌布的亮度是渐变的，这个就比较麻烦了， RGB图像是颜色和亮度混合在一起的，计算机并不容易识别。因此我们需要将图像中的颜色和亮度分离，方法是将RGB转换到HSV。HSV是将亮度、色彩、饱和度分离的，至此我们可以方便的识别颜色并将桌面上的圆选出来了。这时我们再去识别圆形就简单多了。

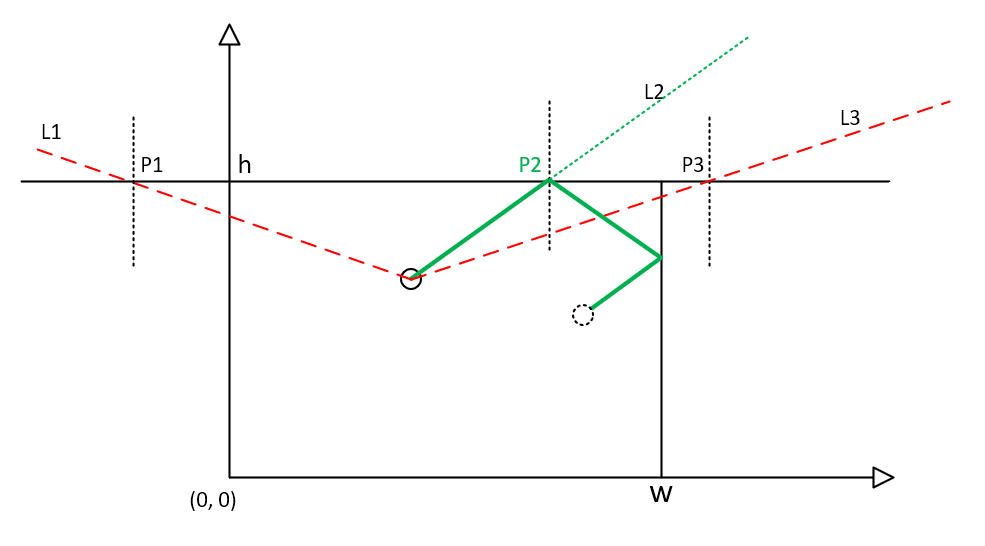
**如何区分：白球、瞄准环、被撞的球以及其他球？**

其中瞄准环是与球区别最明显的，因为它的中心是绿色的。我们是要对每个圆的中心进行采样就能找出瞄准环。其次是白球，明显的是白球和瞄准环有一条较宽的白线，那么我们只要采样一下每个球和瞄准环中心线段的中点的颜色是否为桌布的颜色就可以了。而后被撞的球与瞄准环的距离刚好是球的直径。那么其他的球对于这个简单的辅助也无关紧要了。

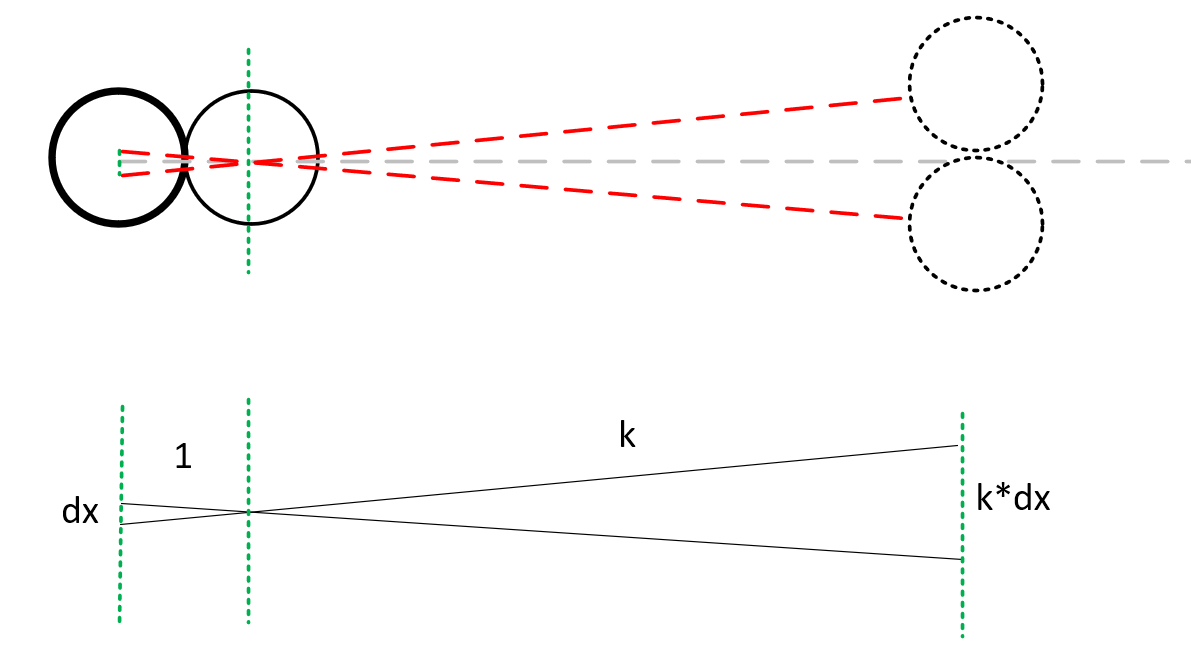
**如何作延长线？**

这个是最简单的部分，以瞄准环为起点，过被撞球做条射线就可以了，如下图的L1、L2、L3（但是这个只能用在把球不经墙面反弹，直接射进球洞的情况）。若是要做球在桌面内多次镜面反射的路径（如图中的绿色线段），这时就需要一些算法了。

具体的做法如下图所示，在已知球的运动方向和距离（这个可以设定）后，我们可以计算出其与四条边的交点坐标，若交点坐标（如P2所示）在四条边上则这一点就是发射点，否则（如P1、P3）就忽略。依次对四条边计算一遍即可找出一个有效的反射点，并且以这条边为对称轴对射线进行对折即可得出反射后的射线。这样循环直到终点在桌面内即可停止。（其间注意当起点在四条边上时，不能计算这条边，否则计算出的还是起点）



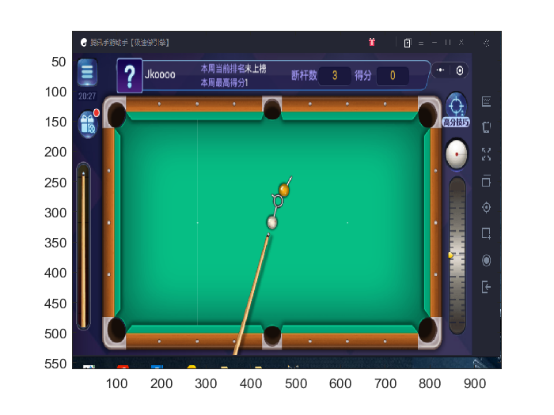
注意：我试过多次发射误差太大，基本不能用。原因是多方面的，首先是圆的位置识别的并不是非常精确，更严重的是通过分析可知：距离每增加一个球的直径就会放大一倍（如下图所示）。这个是非常糟糕的，只能通过更高分辨率的图像来减小误差，但是我的屏幕尺寸不允许，基本无解，除非你直接从游戏的内存空间读取球的坐标。



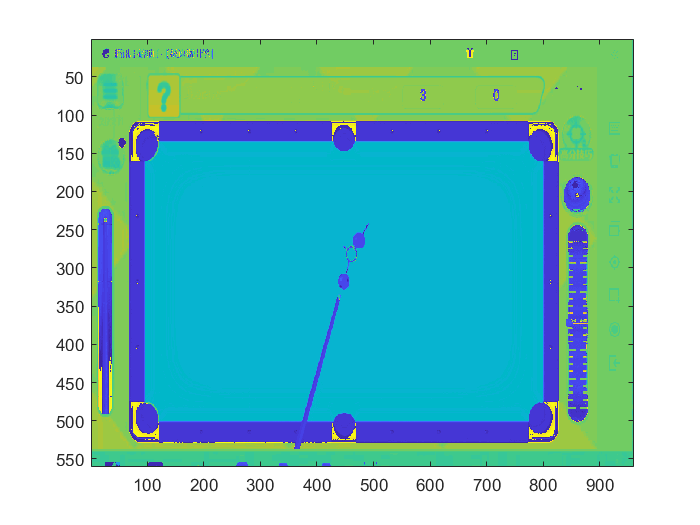
至此整体思路已经介绍完毕，总的来说还是很简单的。

四、实现

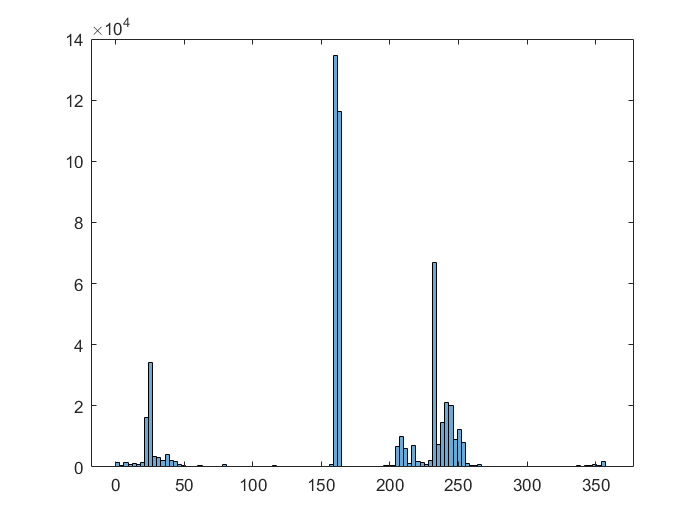
截取图像：



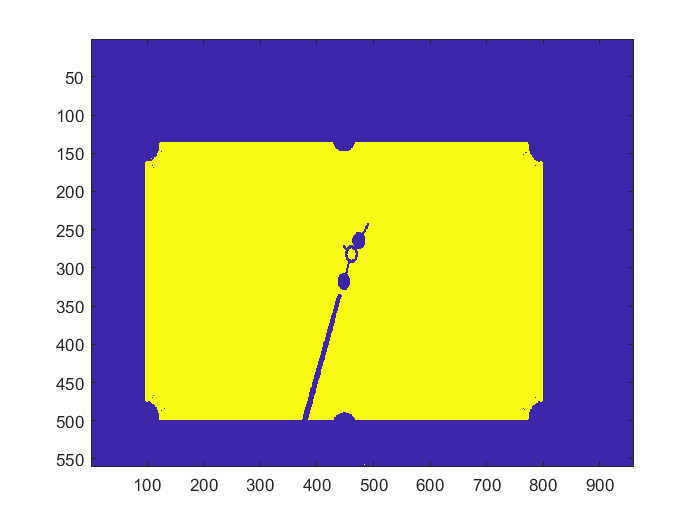
转换为HSV，保留色相



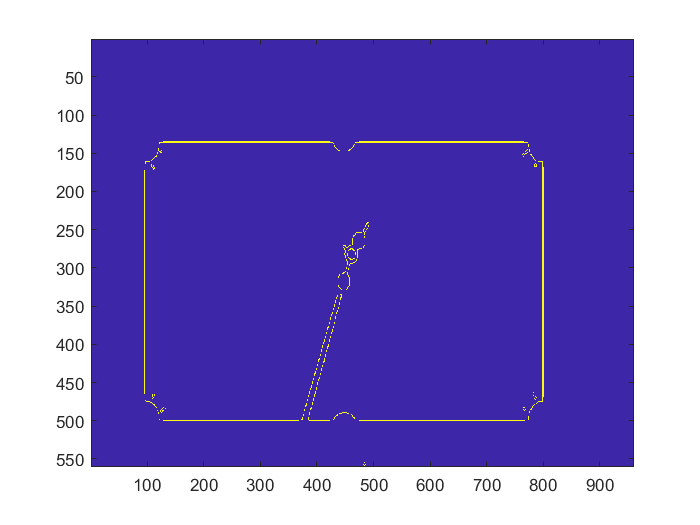
直方图观察桌布的颜色区间



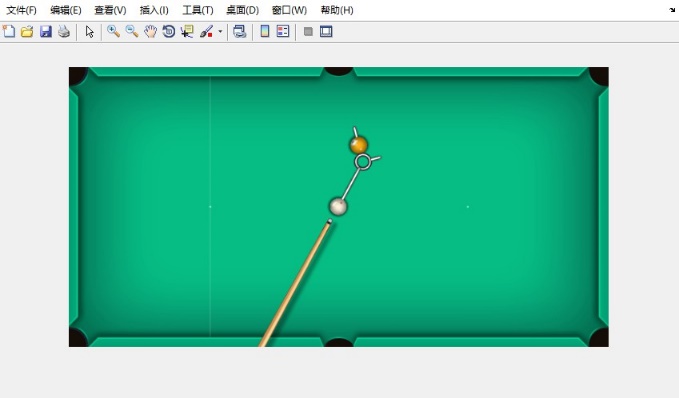
根据颜色区间进行二值化



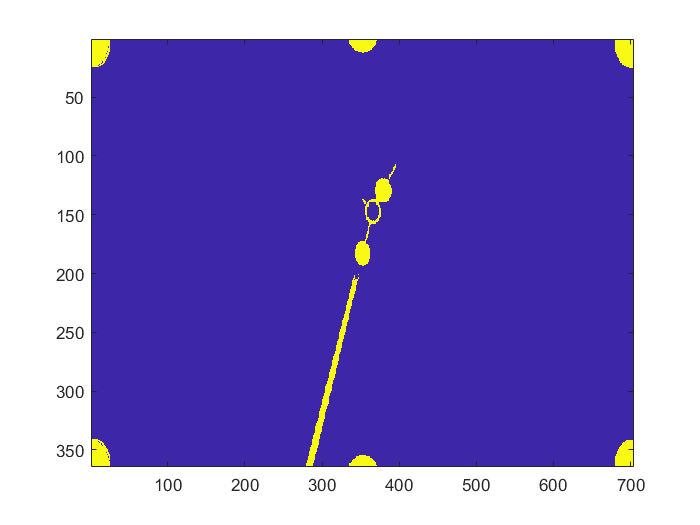
对二值化图像进行边缘检测，为下一步霍夫变换寻找边线做准备，最后根据边线的范围算出球桌的范围。（霍夫变换会对每个像素进行计算，边缘特征提取出来有助于减小计算量）



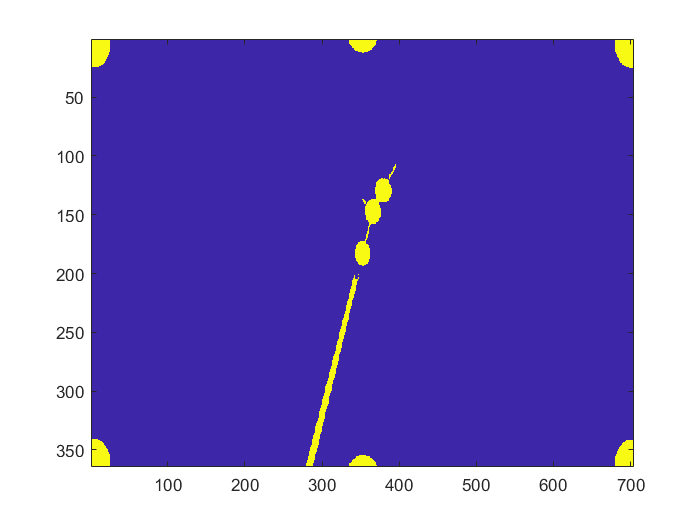
得到桌面的区域，从RGB图像中截取桌面图像



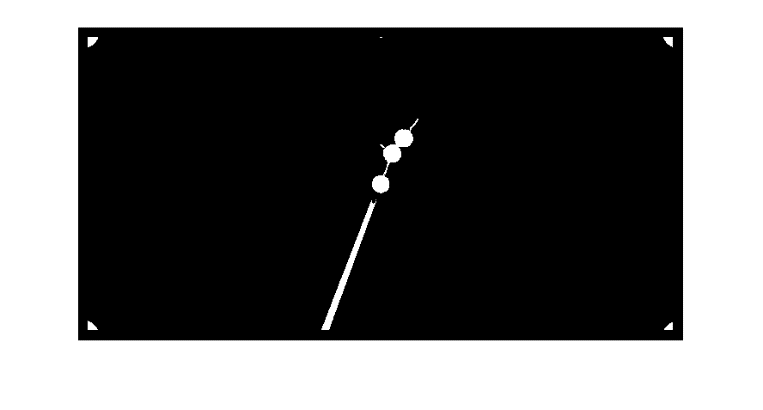
对桌面图像进行二值化，这个图用于判断圆环的位置



对二值化图像进行填充



使用蒙版去除球洞的干扰。之后使用MATLAB内置检测圆形的函数查找球的位置，获取到球的位置后对他们进行分类。



根据分类结果绘出击球的轨迹线

