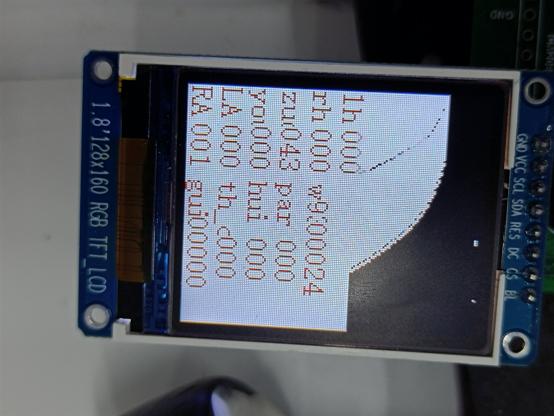
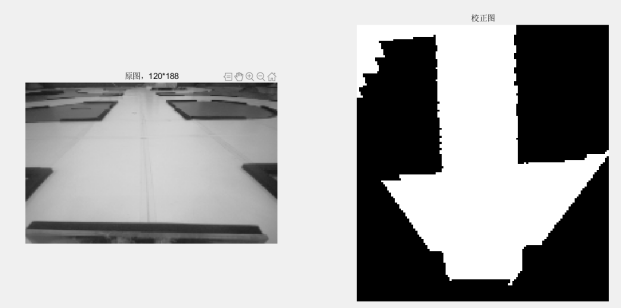
**关于图像丢线补线和小车转弯量不足的思考**

首先解释一下丢线:指搜索左右边界时搜索到了屏幕的边界，比如最右端是188,左是0，那么有右边界搜索到188算丢线，0算左丢线。

下面两幅图 左边这张图右边界全部能找到 左边界全部丢， 右边这张图只是上面的左边界丢了一点。

大部分新手刚开始写代码搜出边界算中线直接是(左边界＋右边界)/2,也不管丢不丢线，就算丢了也是直接(屏幕边界+另一边)/2,但是这样算出的转弯量是和真实需要的转弯量是大相径庭的，非常影响车子的提速和路径，大部分新手转弯调不好也并非转向参数调的不好而是有可能从根源上的得到中线这里就出现了错误。



那如何能找到屏幕的丢掉的一边呢？

我的想法有两个。

先介绍我最推荐的一个，也是我认为最具有理论依据的一个。

1. 这里就体现除了逆透视的作用，首先我们知道我们的赛道宽度实际距离是40cm,半个赛道宽度就是20cm,如果我们直接用逆透视之后的图像来处理丢线那就是叫做一个非常的简单,哪边直接平移40cm就可以得到另一边，这也是我之前强调的逆透视后代码好写的原因之一。

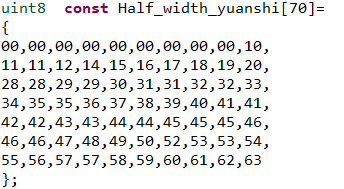
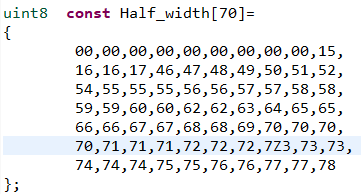
换种思路，如果我们就是想在这种不做逆透视变换的图像中处理丢线的话，我们想补半宽肯定不能补一个定值，因为赛道有畸变,摄像头拍摄到的近处的赛道很宽，远处很窄。

重点来了，所以我们可以我们也可以只将搜能所到的左或右边界来进行逆透视，如果一边丢边，可以将找到的边界进行逆透视后，将逆透视后的边界(逆透视后就是真实的赛道宽度，宽度就是40cm对应的像素点数)来加或减真实赛道宽度得到丢的一边，之后再把处理后的边界再**透视**回来得到丢的那个边界来计算中线，这种方式得到的左右边界使最正确的，最有理论依据的。

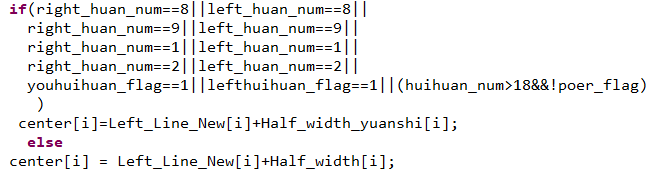
1. 觉得麻烦直接去在原图中手推车模拟一个赛道半宽，当然这是不太准确的，但是相比不管丢不丢线，就算丢了也是直接(屏幕边界+另一边)/2这种方式来对比，它带来的效果也是很不错的。

具体做法是将他们对应的点数写在数组中也行，但是这种方式对应的直道上和弯道上的半宽不一样，普遍情况是直道上半宽小，弯道半宽大。

弯道上的半宽: 直道上的半宽：



最后再启迪一下大家，其实元素的补线也可以用这种加赛道宽度的方式，比如环岛进环之前还有进P回环的那段的补线(维持车头摆正，车走直)，左环岛的话左边界健全，直接用左边界减去赛道宽度得右边界，也省得麻烦得拉线了，出环有那种左右都丢得情况是必须要拉线的，这种方法仅使用于一边找到的情况。



关于有时候车子太偏离赛道导致能用于控制转向行数过少的情况，一般来说只要你图像够大，路径够好是不会有这种情况的，但是既然你提出来了那我可以给个思路，可以在搜索断行上面的行数强行给一个偏离值，既然你车都这种情况了想必已经车偏离中线已经很多了，那干脆给一个打死的量即可。

碎碎念：图像处理只是第一步，代码到最后谁都能写出来，这比赛完赛只是第一步，完赛只是省三而已，想办法跑快才是硬道理。写代码就我觉得和写作文一样，思路框架很重要，不要局限于别人代码，看懂别人代码从来不只是语法上的，而是学习人家好的处理思维从而启迪自己想出更适合自己车的思维，没有一辆车是可以另一辆车的代码不经调试就可以在自己上跑起来的，想太多不如自己实践，别人的方法在你车上不一定适合只是提供一种思路，我想这也是智能车这个比赛最有魅力的一个地方之一吧。

先想到这么多，以后想到有什么重要的再发。