

终点线识别方法研究

闫奕岐

2016. 4. 30

目录

一、直方图反投影及其优化.....	1
概述.....	1
变换颜色空间.....	2
获取直方图前缩减颜色空间.....	3
使用形态学运算.....	4
总结.....	5
其他想法	6
二、均值漂移 (Mean Shift) 算法.....	6
三、连续自适应的 Mean Shift(Cam Shift)算法.....	7
四、识别白色色标	8
概叙.....	8
实现方法与参数调控	9

一、直方图反投影及其优化

概述

直方图反投影是基于像素信息的识别，以下均使用红色色标做实验。色标是我用红色胶带纸制作的。

首先将书上的实例代码不加修改的使用。实例代码采用 RGB 三通道直方图，效果如下：

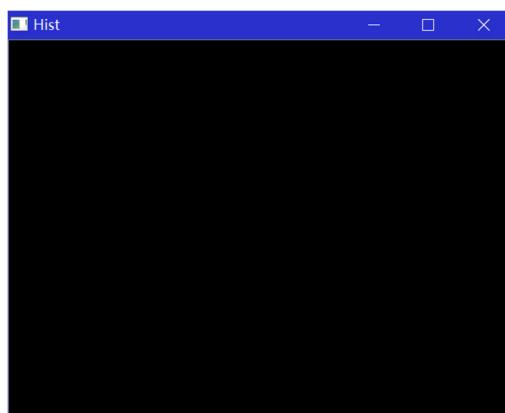


图 1 未经优化的算法

什么都没有识别到。即使是现场采集色标也得不到理想的识别结果。

因为 RGB 并不是基于色彩描述的色彩空间，而且 RGB 通道数值上的差距往往不能反映颜色之间实际的差异，所以变换色彩空间是有必要的。

我还测试了除变换色彩空间之外的其他一些辅助策略，并通过尝试找出了在本次实验中效果比较理想的一种组合策略。

变换颜色空间

HSV 色彩空间是广泛用于计算机图形学中的一种色彩空间，可以用颜色的倒圆锥体表示：

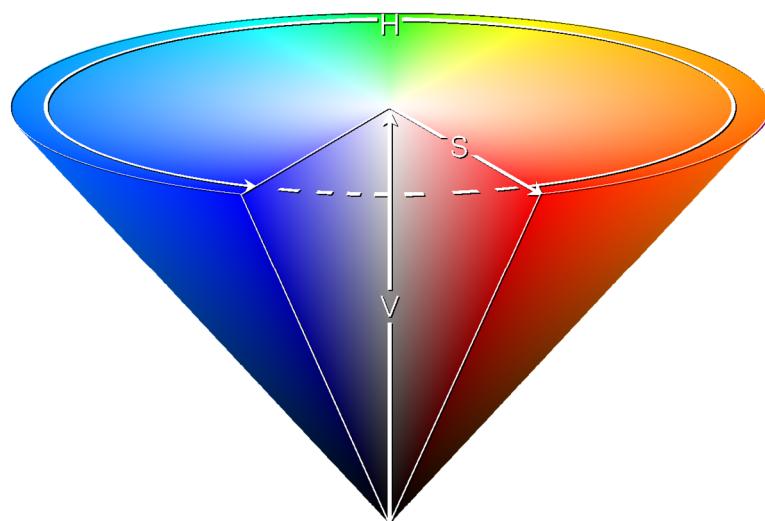


图 2 HSV 色彩空间模型

色调 (Hue) 分量是绕着平行于底面的某个平面环绕一圈的值。显然，Hue 分量能比较恰当的表示不同颜色之间的区别。

然而还有一个不能忽视的问题，就是饱和度 (Saturation) 分量。从上面的模式图中很容易看出，饱和度越小，Hue 分量的“圆圈”越靠近内部（环绕半径越小），各个颜色之间的分辨率越低。比较极端的情况是，饱和度为 0 时，各种色调都集中到椎体的中轴线上，无法区分不同的颜色。因此，在获取直方图和进行反投影之前，我们首先去除低饱和度的像素。通过设定阈值完成这一操作。

事实上，低饱和度像素常常是光照产生的。下面是实际测试中的一个例子：

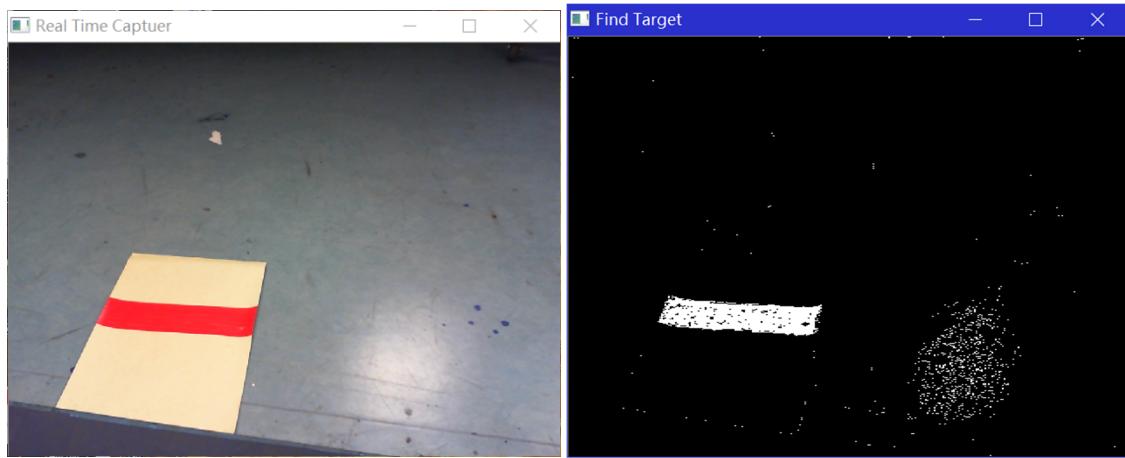


图 3 地板反光产生低饱和度象素

可以看出，地板的反光使图像中产生了一个低饱和度像素区域，直方图反投影对该区域得出了错误的判断。

接下来我们将饱和度分量小于 25 的像素剔除，很好的排除了地板反光的影响。结果如下：

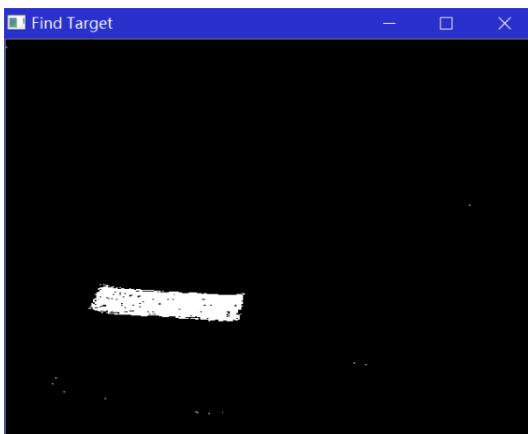


图 4 剔除低饱和度象素

获取直方图前缩减颜色空间

以单通道为例讨论这个问题。试想这样的情况，红色色标规定的标准值是 100，某个像素的值是 85，判别阈值设置成 10，那么这个像素不会被判别为“是红色”，但是实际上我们可能希望 85 也算作红色，只是红的程度弱一些而已。显然，使用完整的颜色空间对于合适的阈值比较依赖。

如果我们适当地缩减了颜色空间，使 100 和 85 划归到新空间的同一个灰度级中，那么判别结果对阈值的依赖程度将不会那么大。

例如，象素值在 80-100 之间的像素都划归为同一年级，那么就算阈值为 0，85 的像素也会被判定为“是红色”。

实际上，即使是完全相同的一块色标，也很难做到百分百匹配。光照条件、

摄像头视角等稍有变化，就会导致计算机“看到”的图像是不一样的。这时，缩减颜色空间可能会有比较好的效果。

必须指出的是，该方法对于使用 RGB 三通道直方图的方法有一定改善作用，对于色调单通道反投影方法是多此一举。由于欠缺更深一步的理论知识，无法解释原因。推测可能是因为色调本身已经能很好区分不同颜色，缩减颜色空间后反而降低了其分辨率。

下图是使用 RGB 空间，缩减颜色前后的结果对比。

显然，缩减颜色空间的效果仍然不及使用 HSV 空间，分辨结果非常不规整。

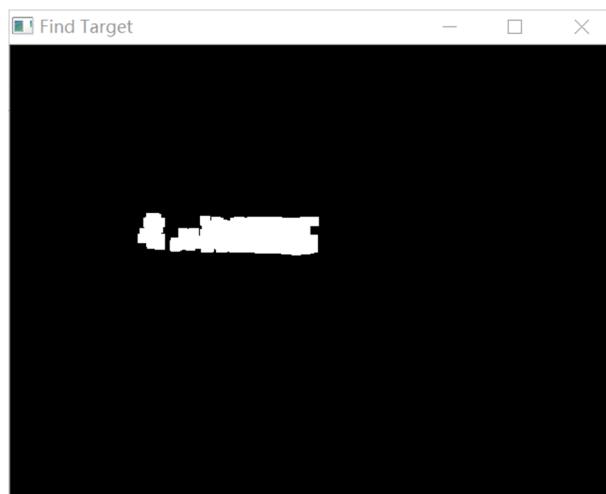


图 5 预先缩减 RGB 颜色空间 使用 RGB 三通道直方图

使用形态学运算

观察图 4 可以发现，识别结果也不是理想的连通域，有很多黑色“孔洞”掺杂其中，这些孔洞可能影响后一步的轮廓提取。使用形态学运算，先腐蚀再膨胀可以很好的去除二值图中的这些“噪点”。

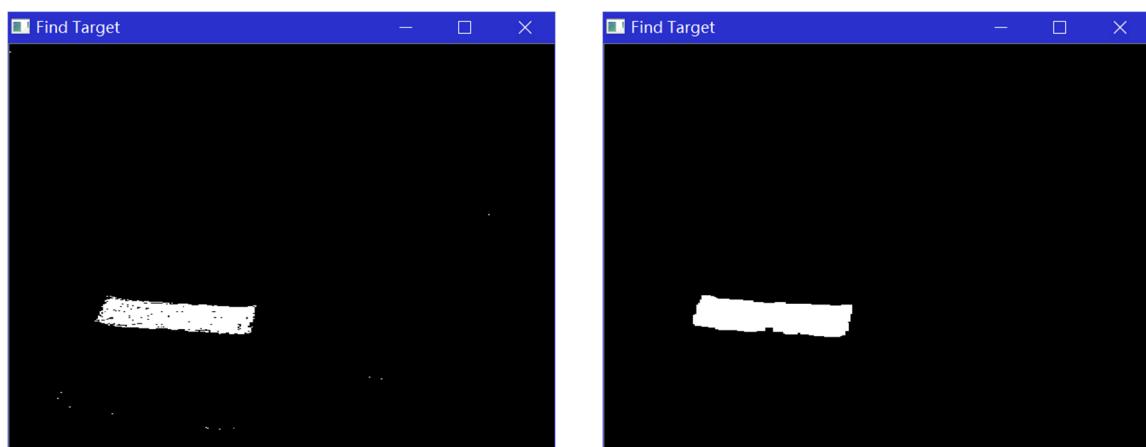


图 5 使用形态学运算去除二值图噪点

总结

1. 完整的策略以及测试结果如下：

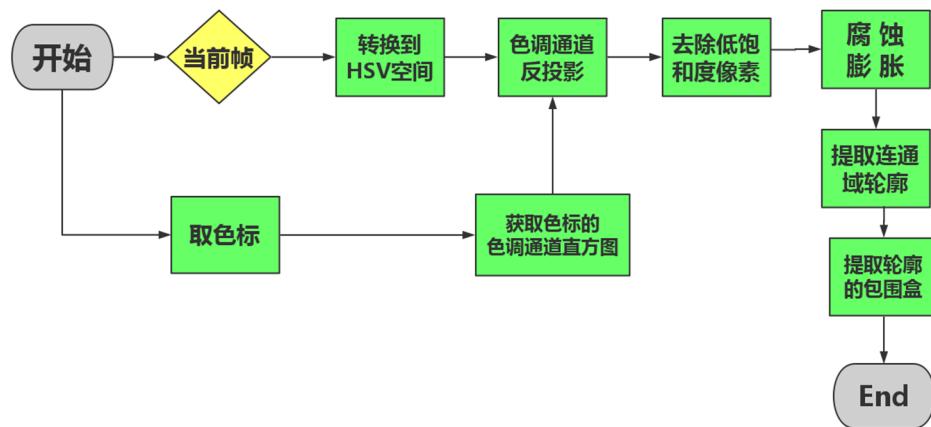


图 6 使用红色色标识别目标的流程

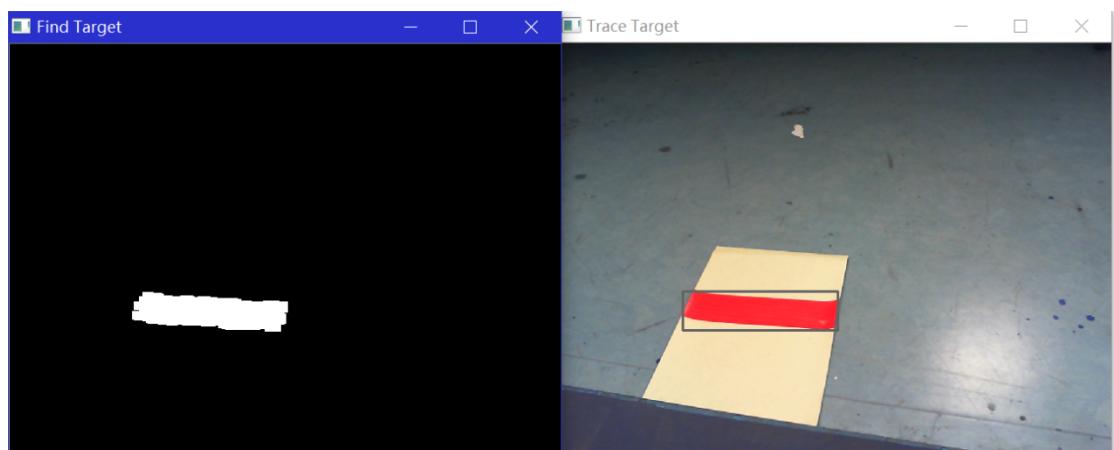
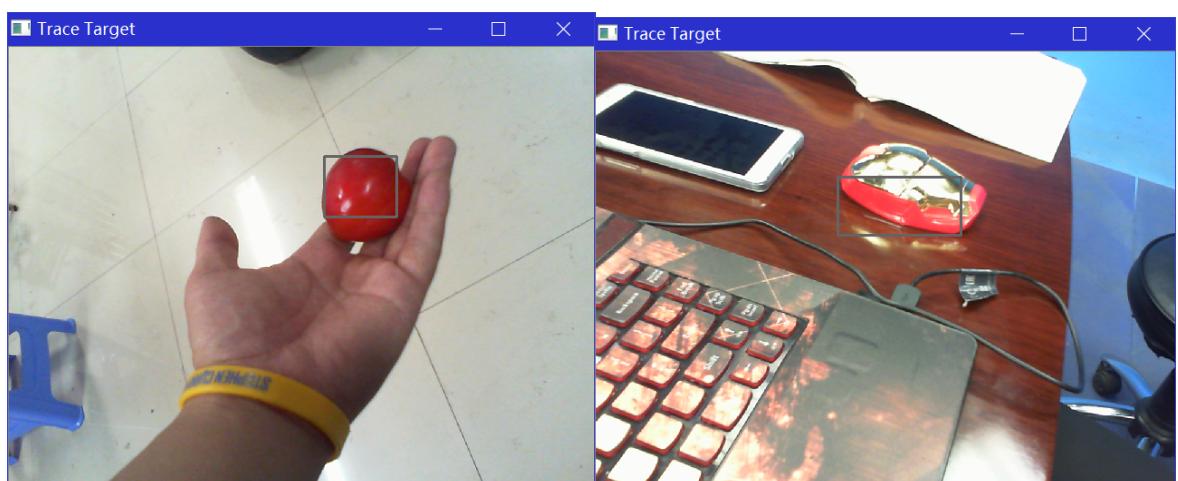


图 7 识别色标本身



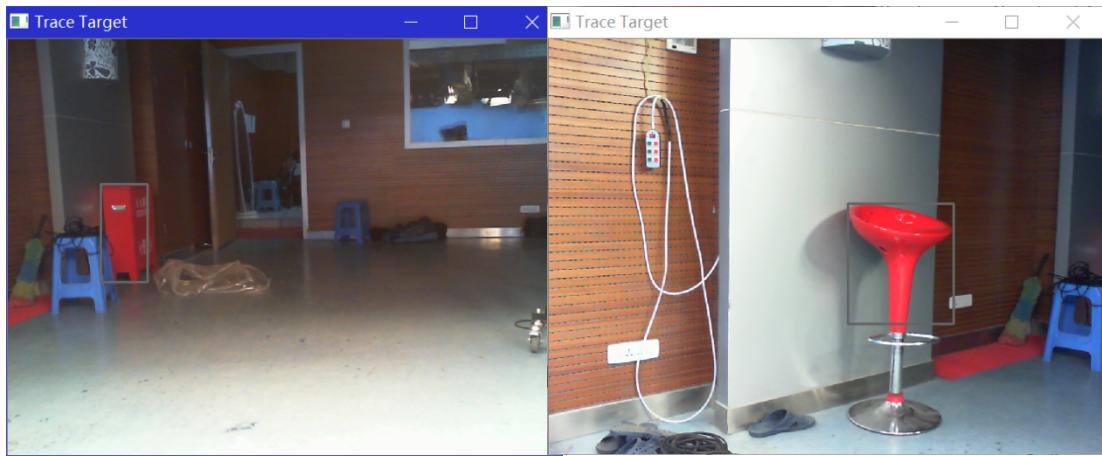


图 8 识别其他红色物体

2. 定制参数

经过优化的算法仍然会受到实际情景的影响。改变环境后往往需要重新调试参数。先将算法执行过程中可以定制的参数列举如下：

- 1) 直方图反投影后的匹配阈值；
- 2) 去除低饱和度象素的阈值；
- 3) 腐蚀膨胀所用的核的尺寸；

其他想法

Lab 色彩空间被设计来接近人类视觉，它致力于感知均匀性。它的 L 分量密切匹配人类亮度感知，因此可以被用来通过修改 a 和 b 分量的输出色阶来做精确的颜色平衡，或使用 L 分量来调整亮度对比。这些变换在 RGB 或 CMYK 中是困难或不可能的——它们建模于物理设备的输出，而不是人类的视觉感知。接下来要尝试使用 Lab 空间，看看是否有更优越的性能。

二、均值漂移 (Mean Shift) 算法

均值漂移算法的原理是在初始位置附近寻找匹配目标，但是搜索只是在初始位置附近进行，目标移动过快就跟踪失败。一旦丢失目标，迭代就会停滞在“跟丢”的位置，只到在该位置附近再出现匹配像素，迭代才能继续进行。

均值漂移算法的另一个缺点是只更新矩形框位置，不更新尺寸，所以目标尺寸变换会导致定位不准确。

实际测试中，手持摄像头移动速度稍快就追踪失败，在轮式机器人不断移动避障过程中，该算法肯定更加不稳定。

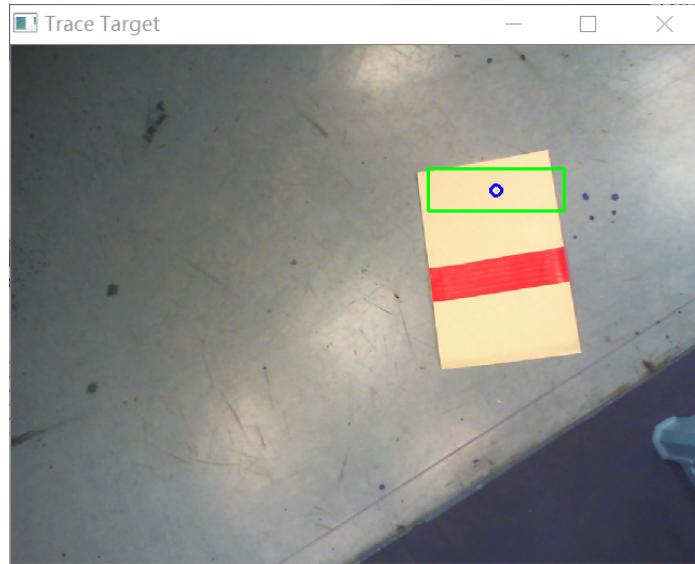


图 9 均值漂移算法跟踪的不稳定性

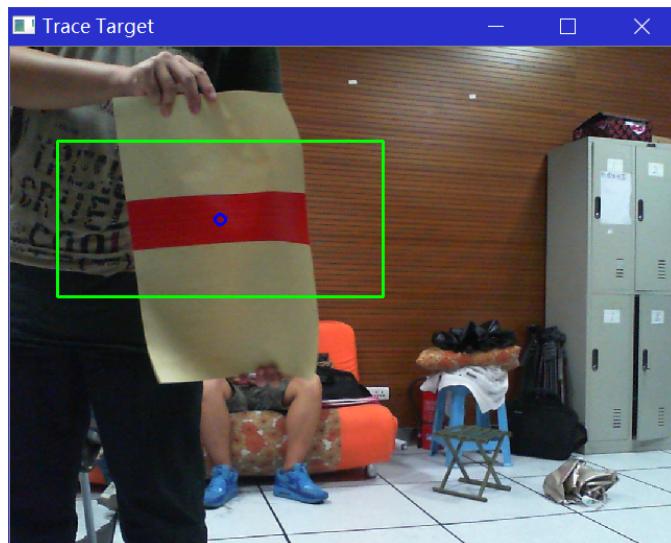


图 10 矩形框保持初始化大小 尺寸不会更新

三、连续自适应的 Mean Shift(Cam Shift)算法

Cam Shift 是对 Mean Shift 的改进，基本思路是连续多次使用 Mean Shift 算法，上一次的结果作为下一次的初值，如此迭代提高精确度。Cam Shift 算法还能动态更新矩形框尺寸。

具体算法实现是：执行一次色调通道直方图反投影算法（见一）提供矩形框初值，然后用 Cam Shift 迭代索搜，得到更新后的矩形框。

有一个容易忽略的问题必须指出。当上一次迭代没有找到匹配像素的时候，矩形框框面积为零，这是再将这个“空”矩形框送入下一次迭代会导致程序崩溃。因此如果检测到空矩形框，需要重新执行一遍初始化。算法流程如下：

跟踪结果比只使用直方图反投影更加稳定，精度也更高。

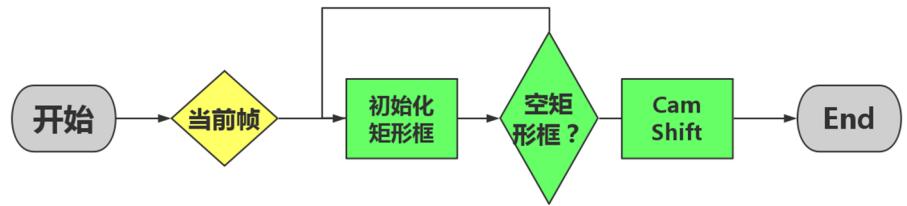


图 11 Cam Shift 算法实现流程

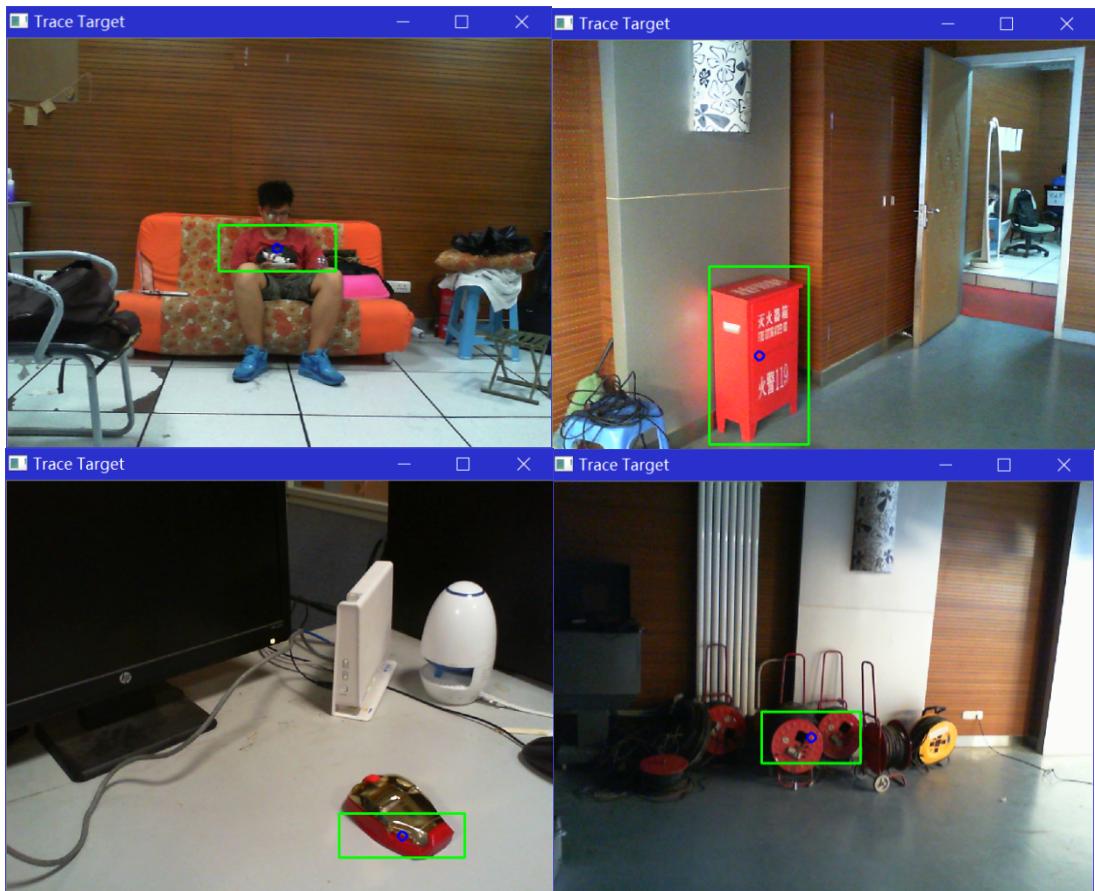


图 11 Cam Shift 跟踪效果

四、识别白色色标

概叙

识别白色的难点主要在于：强光照情况下，反光比较严重的地方也会被误判为“白色”。然而去除低饱和度像素也就一定程度上排除了“白色”，影响目标识别效果。白色的这种特殊性使我们不得不换一种方案。

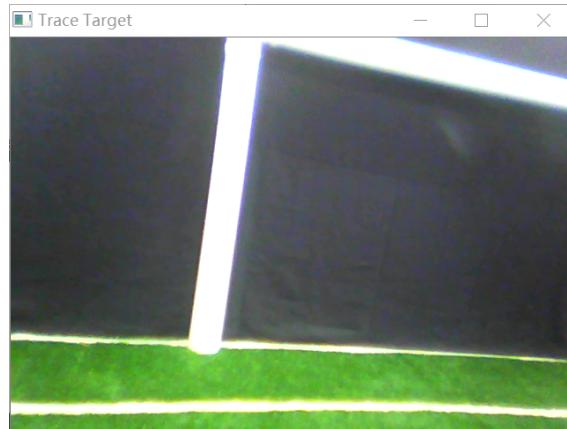


图 12 用识别彩色的方案识别白色色标
没有任何效果

其实新的方案很简单：为什么不使用饱和度通道直方图来进行反投影呢？实际效果表明这种想法是可行的。

实现方法与参数调控

类比识别彩色色标的方法（色调(H)通道直方图反投影，去除低饱和度(S)像素），这次我们采用饱和度(S)通道直方图反投影，去除低明度(V)像素。

去除低明度像素的目的仍然是排除光照影响，周围物体反光的明度一般不会高于纯白色像素的明度。以下实验在光照较强的情景下进行，结果表明，反光较为明显的时候，低明度阈值可能要设置的很大。

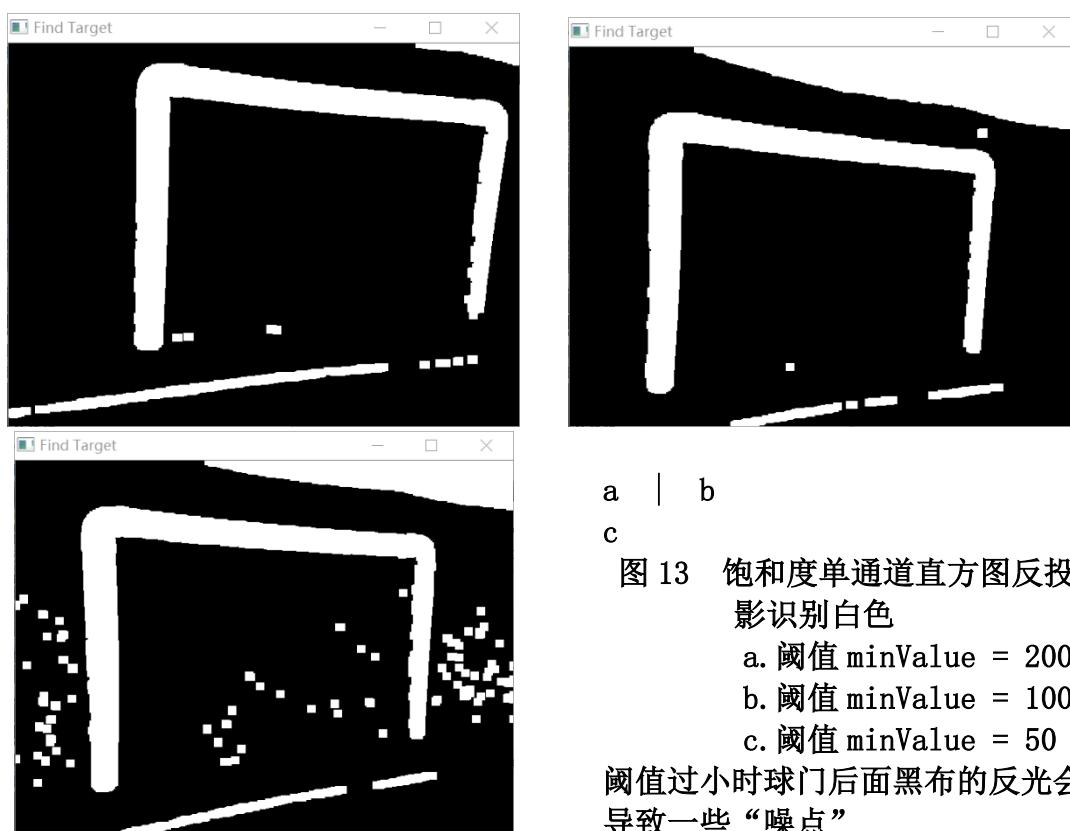


图 13 饱和度单通道直方图反投影识别白色

a. 阈值 $\text{minValue} = 200$

b. 阈值 $\text{minValue} = 100$

c. 阈值 $\text{minValue} = 50$

阈值过小时球门后面黑布的反光会导致一些“噪点”