华南虎视觉组实习生任务

实习生姓名：**钟华**

git仓库地址：https://github.com/1318436194/24-vision-ZH.git

一、公共任务

装甲板识别

装甲板识别效果地址：https://github.com/1318436194/24-vision-ZH/blob/master/%E8%A3%85%E7%94%B2%E6%9D%BF%E8%AF%86%E5%88%AB%E6%95%88%E6%9E%9C.mp4

代码思路

1.图片预处理

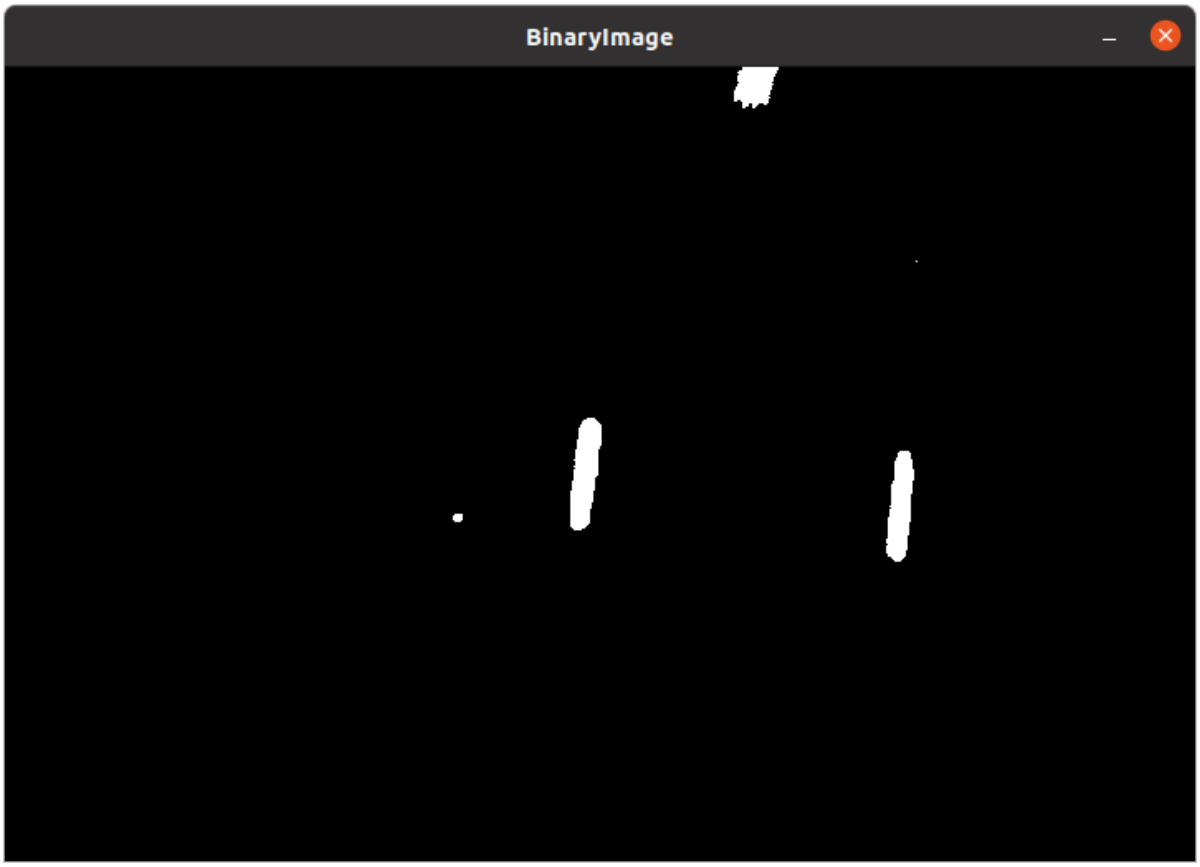
因为视频中灯条红色部分非常明显，与环境相比非常突出，并且没有太多过曝（变成白色），所以直接分离出视频中每一帧图像的红蓝通道，设置阈值参数进行二值化，不进行其他操作

```
vector<Mat>BGRChannels;
Mat BlueChannel;
Mat RedChannel;
Mat BinaryImage;

//分离出红蓝通道
split(src_image, BGRChannels);
BlueChannel = BGRChannels.at(0);
RedChannel = BGRChannels.at(2);

//图像二值化
threshold(light_color=="blue"?BlueChannel:RedChannel,
BinaryImage, 200,255,THRESH_BINARY)
```

得到二值图如下



预处理得到的二值图

如图所示，很好地保留了灯条的轮廓，但是图中还有两个干扰：小光点和上方的**灯管**，需要识别和筛选

2.识别轮廓，初步筛选

用findContours得到最外层轮廓点，并用minAreaRect得到包围轮廓的最小旋转矩形，就得到了待筛选的灯条（一开始尝试过椭圆，但是在我的参数下效果反而不如最小矩形）

设置大小阈值，用旋转矩形的size.area()排除小光点；设置宽高比阈值和旋转角度阈值，排除部分帧中的灯条

3.二次筛选，灯条配对

灯条配对，经过筛选后仍有部分帧中奖将灯管识别为灯条，所以需要进行“灯条”的配对，在匹配的过程中二次筛选。这一步用到了灯条之间的夹角，灯条中心距与灯条平均高度之比，灯条中心竖直差与平均高度之比来排除装甲板和灯管的配对

4.绘制

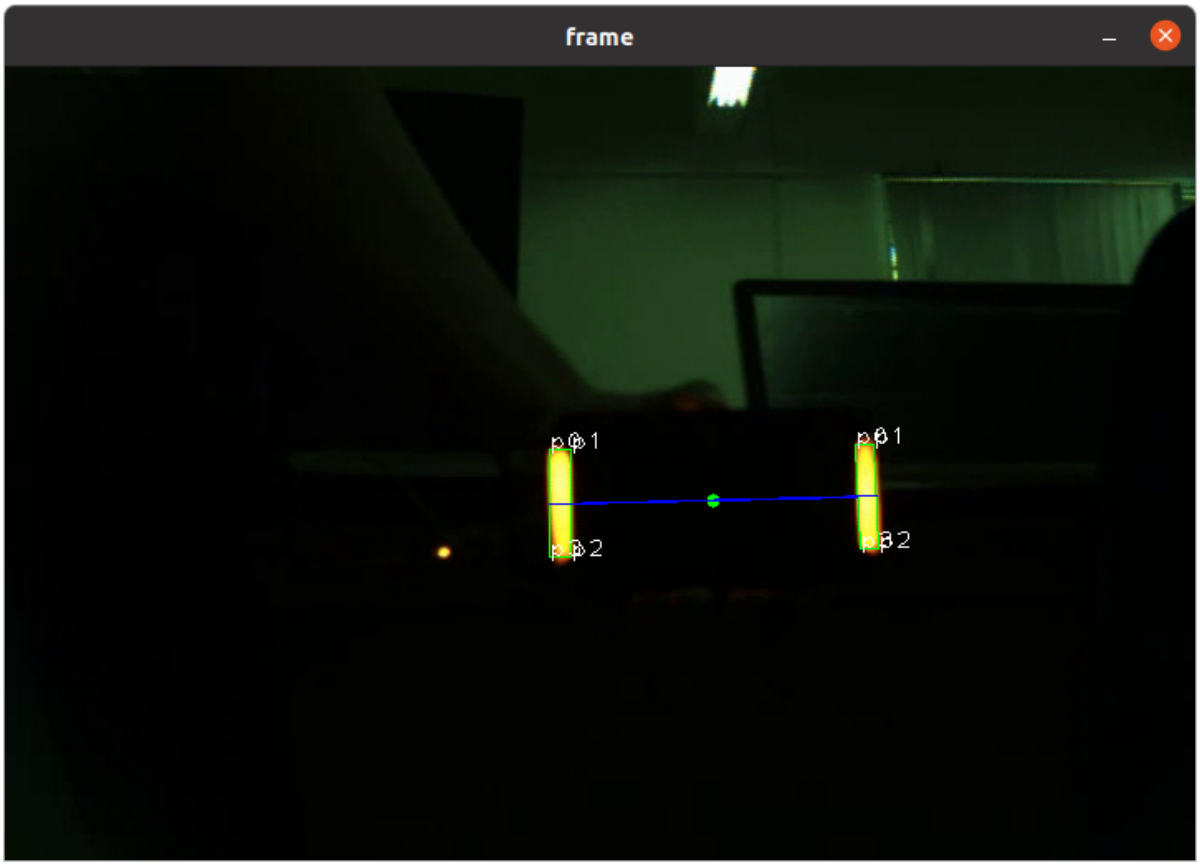
用匹配的两个灯条的四条宽中点作为装甲板的四顶点，进而求得装甲板的中心坐标，并在图中连接装甲板对角线，标出中心点

5.距离姿态解算

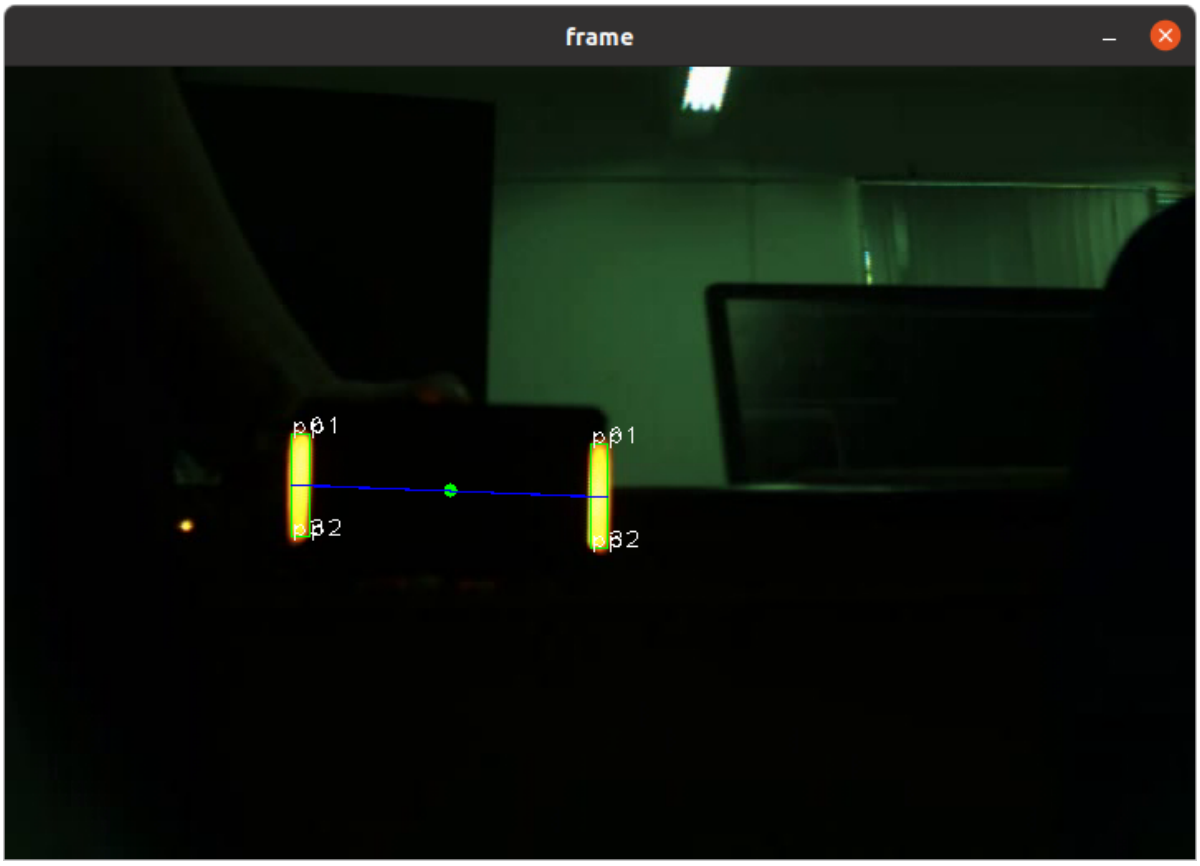
利用SlovePnp解算装甲板的世界坐标，用得到的平移向量计算出装甲板中心到摄像头距离

> ## 遇到问题
>
> **1.OpenCV的RotatedRect太难用😭**
>
> 旋转矩形让我在灯条角度筛选和求装甲板顶点的时候破了大防

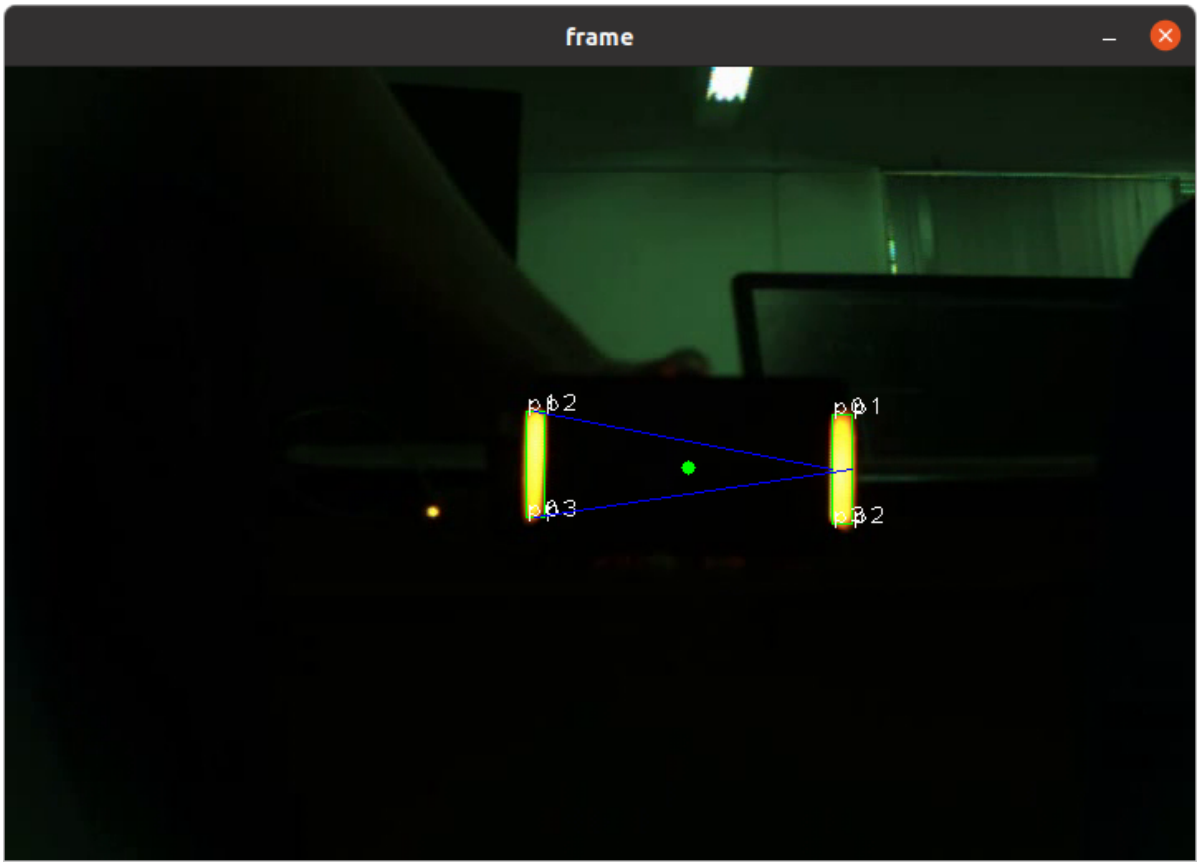
RotatedRect.points(pts)得到的四个点顺序经常抽疯，官方文档说以左下角点(bottom left,bl)为pts[0]，但是实际上minAreaRect得到的旋转矩形并非如此😓拿图说话：



0点在左上角



0点在左上角



0点一个在左下角，一个在左上角

因为我的算法是用灯条矩形的0，3点中点，和2，4点中点作为装甲板的顶点（这样是由于官方文档说第0点是左下角点），所以当旋转矩形的点顺序抽风成上面3图时，本来连的对角线就会变成各种抽象连线😿

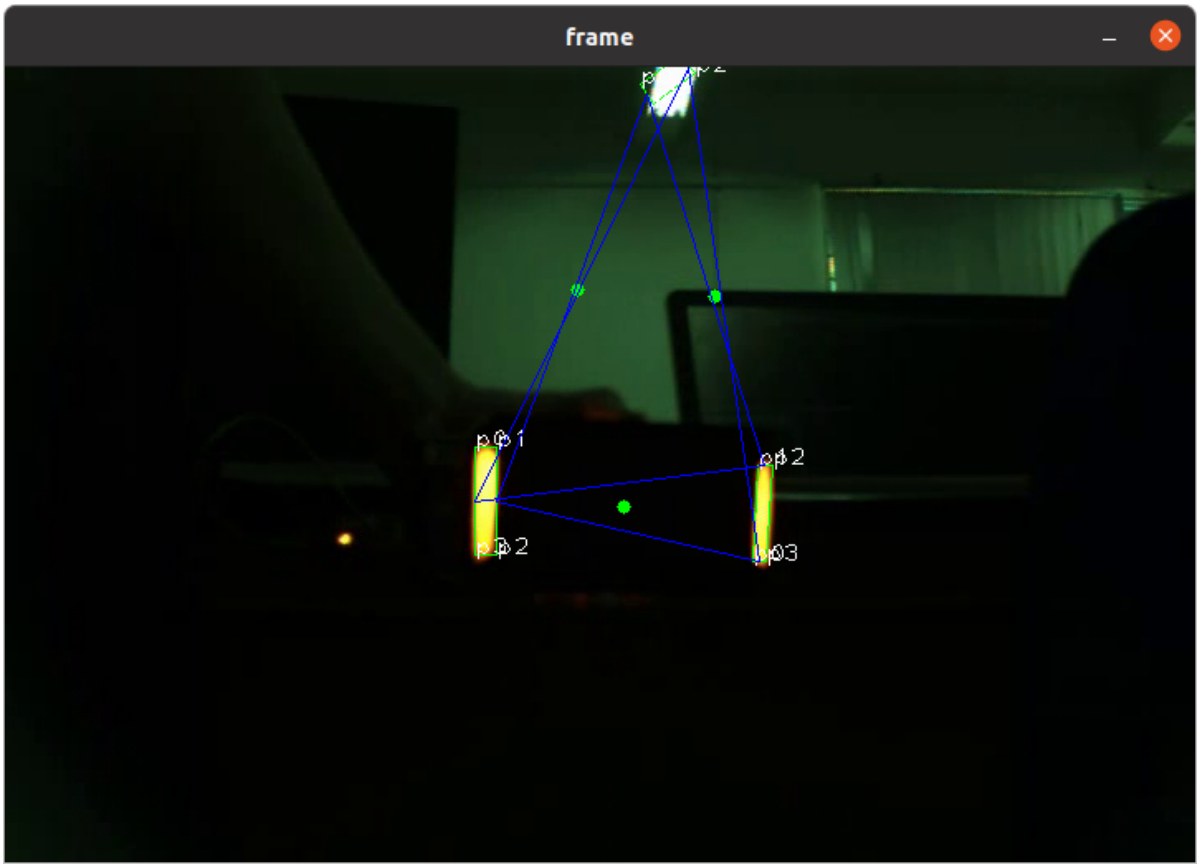
最最最破防的是一开始我对官方文档深信不疑，所以画出上面的图时我绞尽脑汁也不知道是哪里出了问题，还有角度也跟着一起出问题，因为旋转矩形的angle是以x轴顺时针旋转到0，3点构成的边的最小角度计算的，当0点抽风到tl，br位置的时候，灯条角度差就会出大乱子，直接爆涨90度，一下子让灯条被我的识别和筛选算法排除。发现是旋转矩形出了问题时，我反复改代码试了很久，**妄想**通过发现旋转矩形顶点的规律来使用它，结果只能是被现实压垮

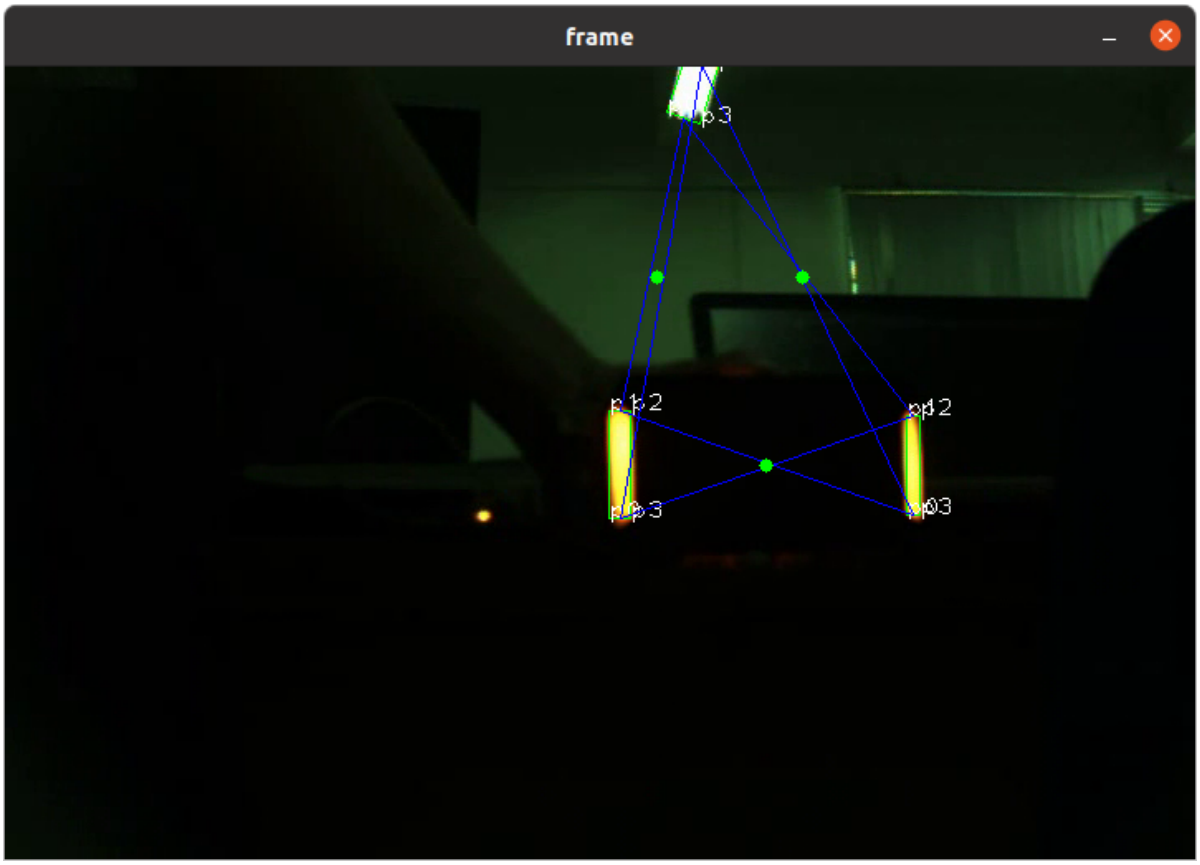
2.调参调到心态爆炸💥

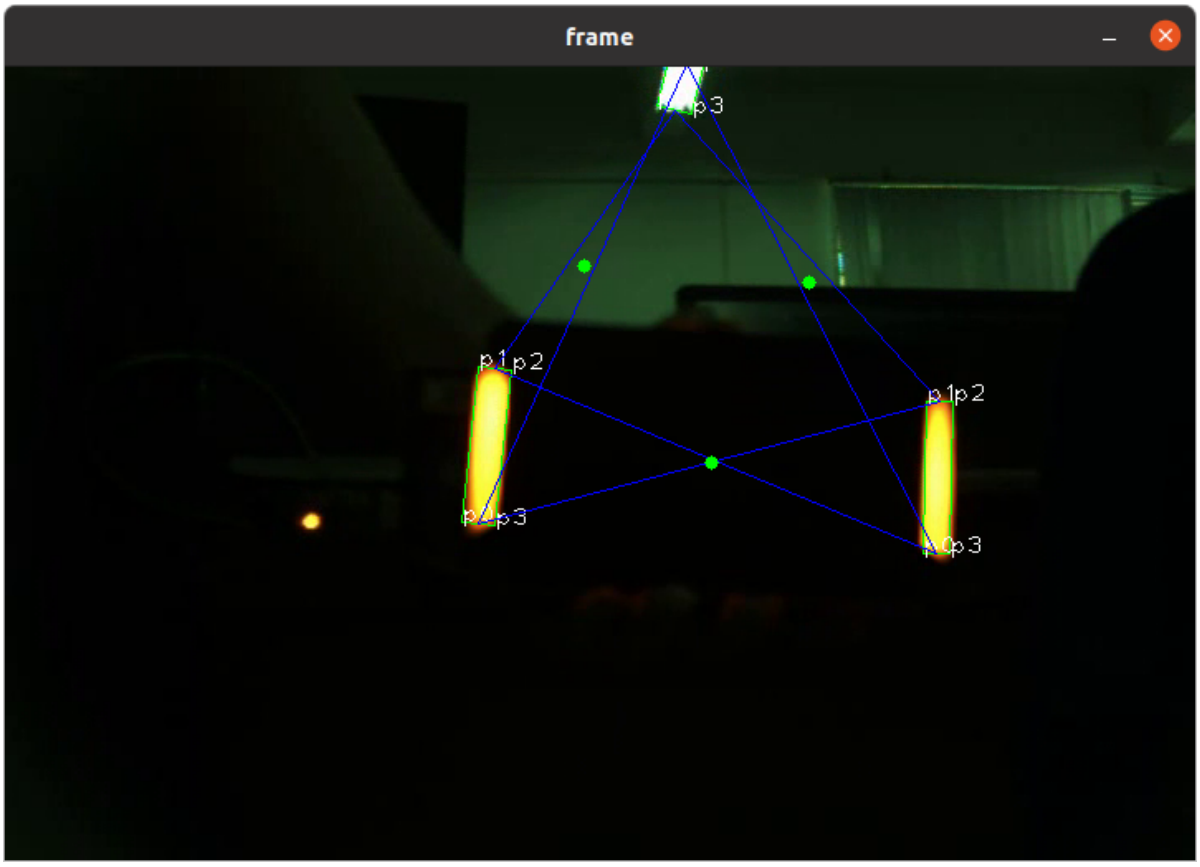
一开始从网上的各种开源代码中寻找灯条筛选思路，但是给定的视频并不能完全套用开源代码中的参数，而且一开始并不知道装甲板的实际尺寸，所以经历了一个$\begin{matrix}\text{痛苦?}\\ \text{快乐}\surd\end{matrix}$的调参过程

那种刚调好参数，然后想试试换个算法识别轮廓，再跑一次直接一个灯条都识别不出来的心情真的.....

附上因为参数没调好，把灯管也识别成灯条的效果图：







有没有一种~~天女散花~~不知死活的美

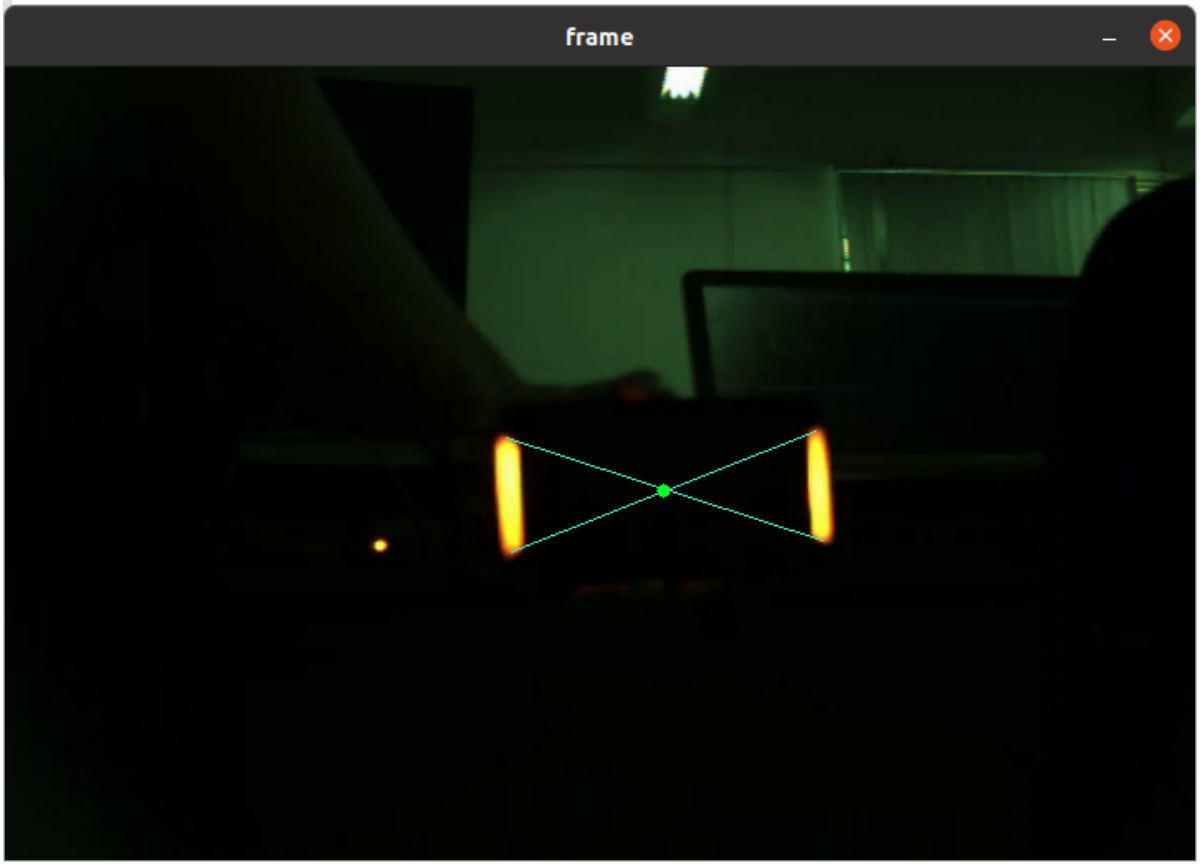
> ## 解决思路
>
> 1. 自建了一个my_rotatedrect结构体和从RotatedRect得到my_rotatedrect的静态函数to_my_rorec。在函数中求出左下角点作为第一点，并将旋转角度，面积，尺寸，宽高比传入my_rotatedrect，这样就可以放心食用RotatedRect了

```
struct my_rotatedrect{
        Point2f center;             //旋转矩形中心点
        float angle;                //旋转矩形的宽
        Point2f points[4];          //4个顶点，以左下角为第一个点，顺时针顺序
        float width;                //以点0和点3构成的边为宽
        float height;               //以点0和点1构成的边为高
        float aspectRatio;          //宽/高
        float area;                 //旋转矩形的面积
    };
```

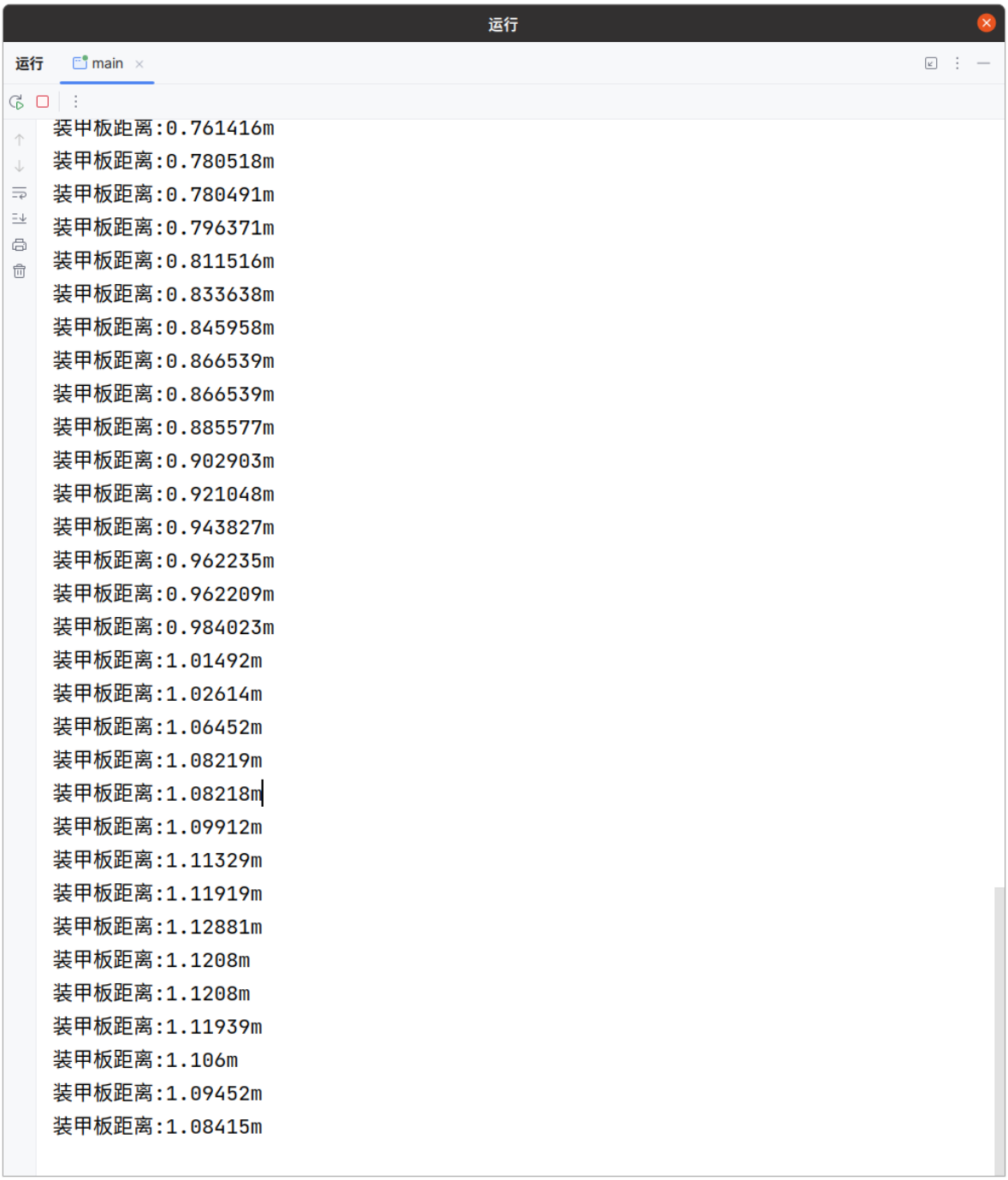
```
10
11 void ArmorFit::to_my_rorec(const RotatedRect &rect,
   my_rotatedrect& myrect) {
12     Point2f ps[4];
13     rect.points(ps);
14     int bl_infex;
15     float bl_x=100000;
16     for (int i = 0; i < 4; ++i) {
17         if (ps[i].y>=rect.center.y &&ps[i].x<bl_x){
18             bl_x=ps[i].x;
19             bl_infex=i;
20         }
21     }
22
23     for (int i = 0; i < 4; ++i) {
24         myrect.points[i]=ps[(bl_infex+i)%4];
25     }
26
27     if (bl_infex!=0){
28         myrect.width=rect.size.height;
29         myrect.height=rect.size.width;
30         myrect.angle=rect.angle-90;}
31     else{
32         myrect.width=rect.size.width;
33         myrect.height=rect.size.height;
34         myrect.angle=rect.angle;
35     }
36
37     myrect.aspectRatio=myrect.width/myrect.height;
38     myrect.area=rect.size.area();
39     myrect.center=rect.center;
40 }
```

2. 发掘了Clion这个巨好用的IDE，不用在命令行中输输输，直接就可以很方便地运行和调试，和VS Code相比，需要自己配置的东西很少，代码补全和参数提示也很更加智能，Git也是自动配好不需要额外下载插件就可图形化使用;然后就是将经常需要改动调试的参数写到配置文件中去，调试直接在配置文件中改，就会方便很多

效果图



装甲板识别效果



装甲板中心到摄像头距离输出效果图

总结

1. 图像预处理的过程还是不到位，因为滤波还没有学得很好，所以需要加强这方面的学习
2. 单靠传统视觉识别还是有上限，需要深度学习方面的知识

3. 多文件编程真的很好用（不知道我理解的对不对哈哈哈哈哈）。将程序的结构按功能分成多个源代码实现，可以提高复用性，在程序很长的时候要改对应的功能直接去对应的源文件改即可，避免一个冗长的总程序代码看得眼花缭乱。如果将参数保存在配置文件中，在程序中读入，在调参的时候，就可以大大省去在每一处代码中改参数的步骤
4. modern C++需要深入了解，模版和泛型编程很有意思，还有STL库.....太多了，慢慢学吧😭
5. Clion yyds！！！（目前还驾驭不了高度自定义的VS Code.....）
6. 万事开头难，虽然过程艰辛，但是完成效果的那一刻，我感觉一切付出都值得
7. 华南虎，不要怂，就是干！！！

二、专属方向哨兵任务

1. 代码思路
2. 遇到问题
3. 解决思路
4. 效果图
5. 总结

三、总结

运行效果地址：https://github.com/1318436194/24-vision-ZH/blob/master/%E8%A3%85%E7%94%B2%E6%9D%BF%E8%AF%86%E5%88%AB%E6%95%88%E6%9E%9C.mp4

git仓库地址：https://github.com/1318436194/24-vision-ZH.git