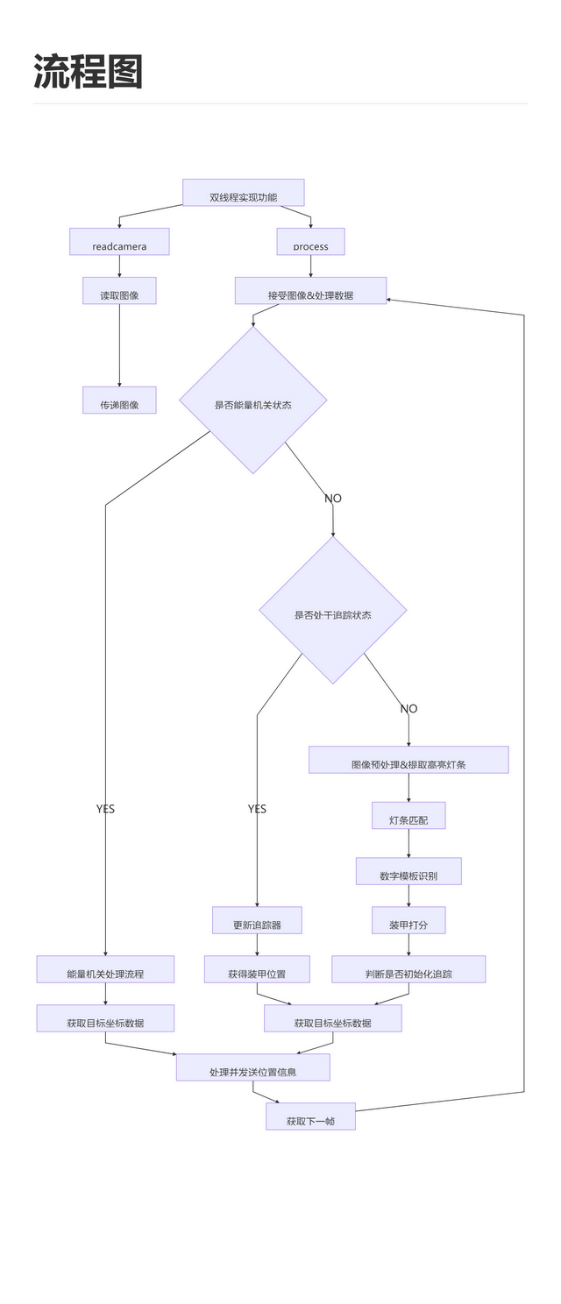
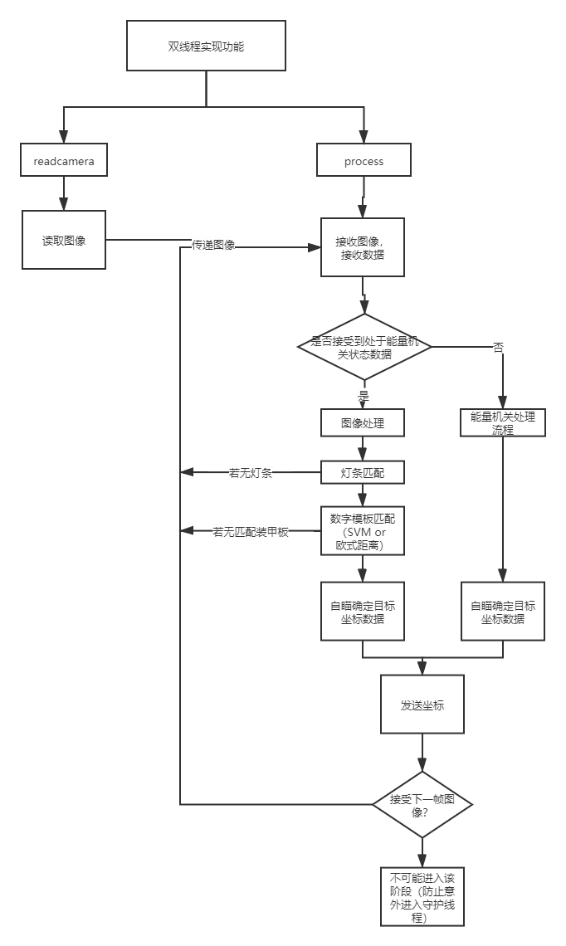
**视觉组2020**

1. **总工程介绍**

本工程目前主要实现了以下这些功能：装甲识别、能量机关识别预测、数字识别、目标跟踪



左：总工程流程图

右：加入目标跟踪后的总工程图

详情可查看https://github.com/tsagaanbar/muc-rm-2020/blob/master/docs/intro.md

工程还采用通过配置文件读取设置参数的功能，方便调试程序；

1. 算法介绍

* 装甲识别

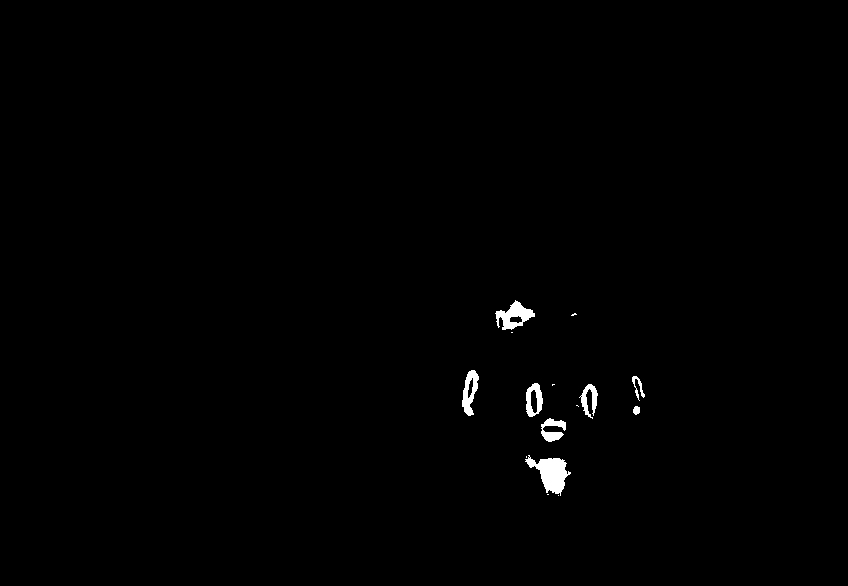
该部分的步骤和思路如下：

1. 图像预处理，进行开闭运算，获取三种输入图像

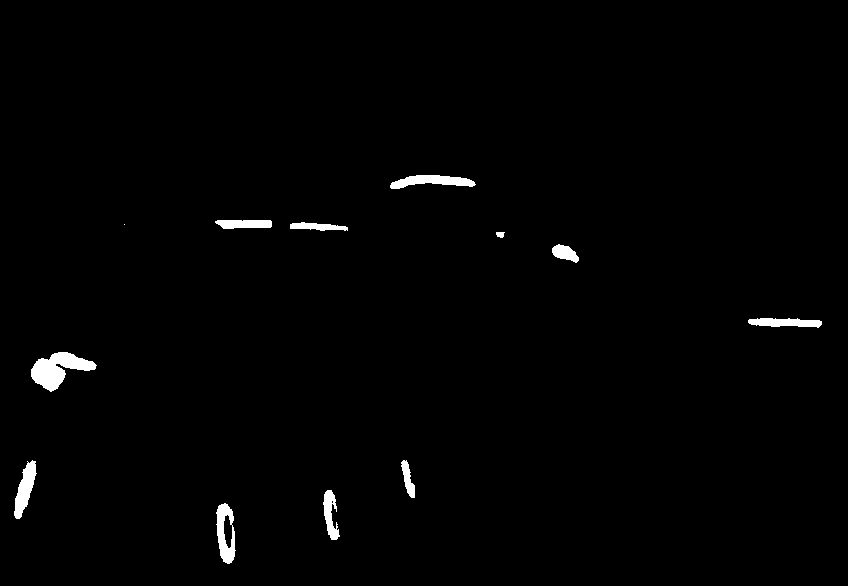
例图：

原图、蓝色通道B、红色通道R

**蓝色通道B - 红色通道R（检测蓝色）、归一化并二值化**

**红色通道R - 蓝色通道B（检测红色）、归一化并二值化**

1. 获取部分轮廓，并使用过程图区分红蓝。

步骤1中得到灯条的颜色轮廓，该步骤需要判断“灯条”是否处于颜色轮廓内，若是则写入结果

1. 通过算法，快速找结果灯条：
2. 查找轮廓
3. 循环遍历所有轮廓，调整轮廓角度信息，进行信息归一化处理。
4. 对长宽比例通过参数进行一次筛选。
5. 将结果返回到vector变量中。
6. 使用装甲检测，调用数字分类器进行分类。
7. 遍历所有灯条进行匹配（通过各类参数，包括长宽比限定，长长，宽宽限定，角度差值限定。）
8. 对已通过筛选灯条进行装甲组合，并赋值（可优化）。
9. 对结果使用分类器进行分类（如SVM，欧氏距离）。
10. 对分类后装甲板返回结果。
11. 对分类结果进行打分（装甲打分，择优攻击）
12. 做出最终选择（考虑重力补偿）。
13. 返回装甲结果。

* 目标跟踪

在未采用目标跟踪时，算法通过对逐帧图像进行识别从而检测出装甲板；然而该法对光照条件等要求高，有可能识别失败，故采用目标跟踪来弥补相关问题；

但目标跟踪一般适用于平滑移动或垂直视线平面内旋转的物体效果较好，但对于实战中的“小陀螺”，即在三维空间中旋转的物体，目标跟踪并不非常适用。仅当旋转速度为特定值时，能实现“伪跟踪”。

本工程内采用ECO算法（调用公开的该算法C++实现），本质上为检测式目标跟踪，和KCF相同，可以算是其衍生版本，但具有可变尺度，且因为其对样本有做分类等，跟踪效果较KCF提升明显；缺点是速度问题，在初始化阶段会耗费很长时间，同时跟踪帧率也远低于KCF。

理论上，追踪器可以追踪视频中的任何物体，故本模块采用跟踪的目的并非缩小检测范围，而是直接追踪目标位置。

程序认为，识别到装甲板并选择出最佳选择后即可开始尝试生成用于跟踪的bounding box；

通过包围装甲板的旋转矩形生成合适大小的bounding box，即需要包裹住整个装甲板以获取足够特征，又不能太大而导致跟踪一段时间后中心偏移；

然后判断生成的bounding box是否合适，若不合适，如比例失当、面积过大过小、最小包围矩形与装甲板面积相差过大，则不认为能够开始跟踪模式；

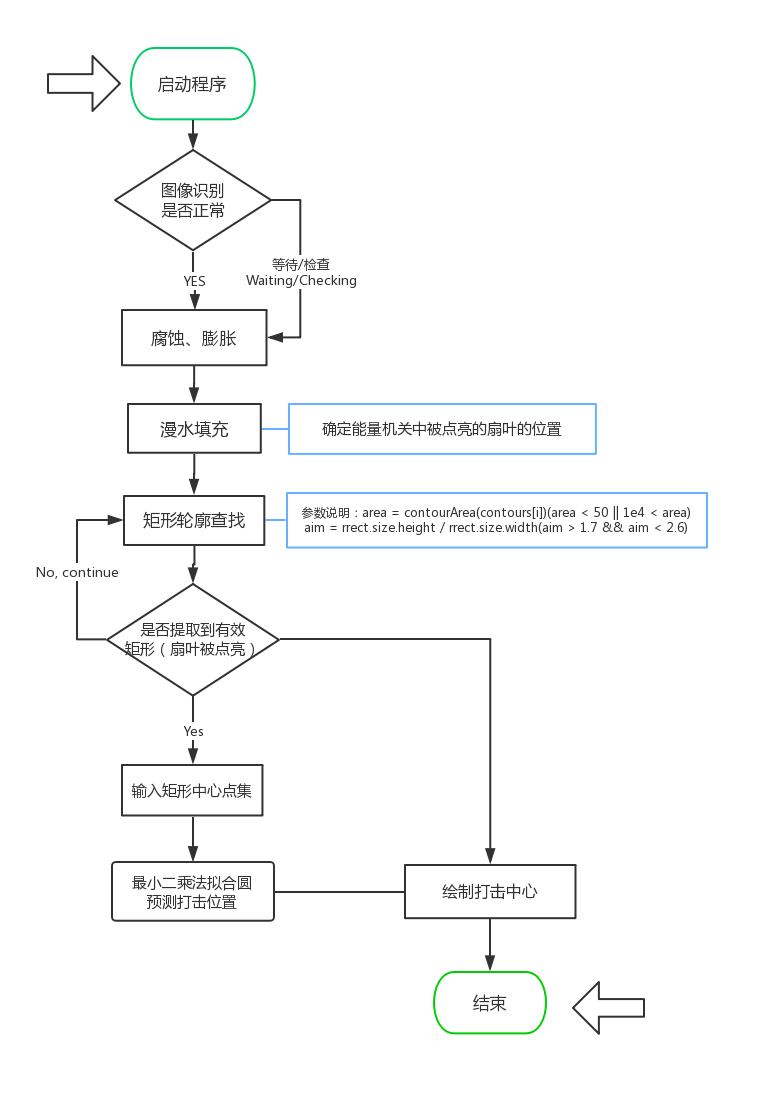
若合适，则初始化跟踪器，并将之后接收到的视频帧交给跟踪模块进行识别。

算法初始化和更新时接受普通的矩形框，而非前阶段检测算法生成出的旋转矩形，在包裹准确度上会有所下降；但算法大部分时间能保证追踪框中心处于装甲板中心处，基本上不影响生成用于发射的坐标；

为了避免追踪时间过长带来的问题，追踪部分将在跟踪一定帧数后退出，或在检测到应当退出的情况时退出，如追踪框比例失当、追踪框过大等情况。

工程预备设计其他备用跟踪方案，如：采用简单的角度预测、卡尔曼滤波、或者预测型目标跟踪算法。

* 能量机关识别



程序框图

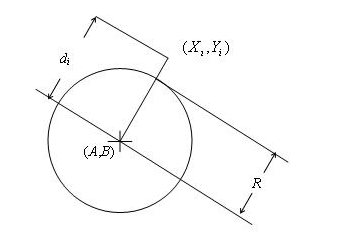
拟采取的算法——最小二乘法拟合圆

在对能量机关进行矢量化的过程中, 我们尝试了多种常规的拟合方式。比如在理论研究中提到最多的利用折线段进行拟合，但在实际使用中，用直线对能量机关扇叶的运动轨迹曲线进行拟合只能大致反映曲线的方向，不能客观地反映曲线的轮廓特征, 而且用这种方式拟合形状精度不高，难以进行下一步的预判。

针对上述问题，我们决定将直线拟合与圆弧拟合相结合，进行了充分的理论研究并且找到了一种较好的解决方案。采用混合圆弧-直线逼近法来逼近能量机关的原始轮廓，用样条曲线拟合轮廓曲线，来达到定位能量机关并进行预判的目的。我们尝试了各种拟合圆方法，如基于遗传算法的相切圆弧逼近曲线算法，利用垂直平分线求交法对曲线进行拟合的方法, 以及利用圆内接多边形的任意一条线段都通过圆心的原理，将满足圆弧拟合条件的折线段进行圆弧拟合等多种方法。最后，我们综合比较各种方法在实际操作中的可行性，决定采用最小二乘法对圆弧进行拟合，通过最小化误差的平方和找到图形采样点的最佳函数匹配，利用这种算法得到的能量机关圆弧，是采样点的最优解，即最佳逼近圆。

该算法的核心思想是：获取可以拟合为圆弧的能量机关折线段顶点坐标数据，根据误差平方和最小化原则, 找出其匹配的拟合圆函数。

算法原理



圆的方程为:

其中圆心坐标为 (A, B)，半径为R.

对圆方程进行变换, 得到另外一个形式, 如式（1）所示:



令：

得到圆方程的简化形式, 如式（2）所示:



在式（2）中，只需要求出未知参数a，b，c就可以求得圆心坐标及半径，如式（3）：



设对能量机关中被点亮扇叶采集的点样本集为(Xi, Yi)，点个数为N，样本集中各点到圆心距离的平方与半径平方的差为:



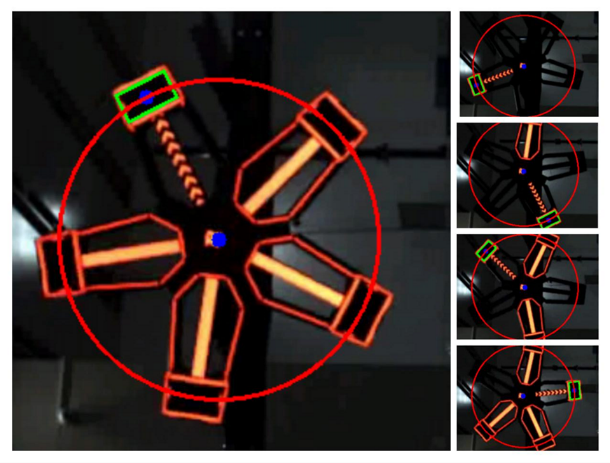
利用最小二乘法拟合圆即求得参数a, b, c, 使得平方和

为最小。

F(a,b,c)分别对a,b,c求偏导，令偏导为0，即得到极值点，如式（5）所示：



解此线性方程组求得参数a,b,c后，代入式（3）中，即可得到圆心坐标 (A, B) 及半径R的值。



效果图

* 数字识别

主要有两种方式：（无流程图）

**方式一：SVM模板匹配**

Ptr<ml::SVM> svm = Algorithm::load<ml::SVM>(SVM\_PATH);

HOGDescriptor\* hog = new HOGDescriptor(Size(40, 40), Size(20, 20), Size(10, 10), Size(10, 10), 9);

//HOGDescriptor\* hog = new HOGDescriptor(Size(9, 9), Size(6, 6), Size(3, 3), Size(3, 3), 9);

//HOGDescriptor\* hog = new HOGDescriptor(Size(6, 6), Size(6, 6), Size(3, 3), Size(3, 3), 9);

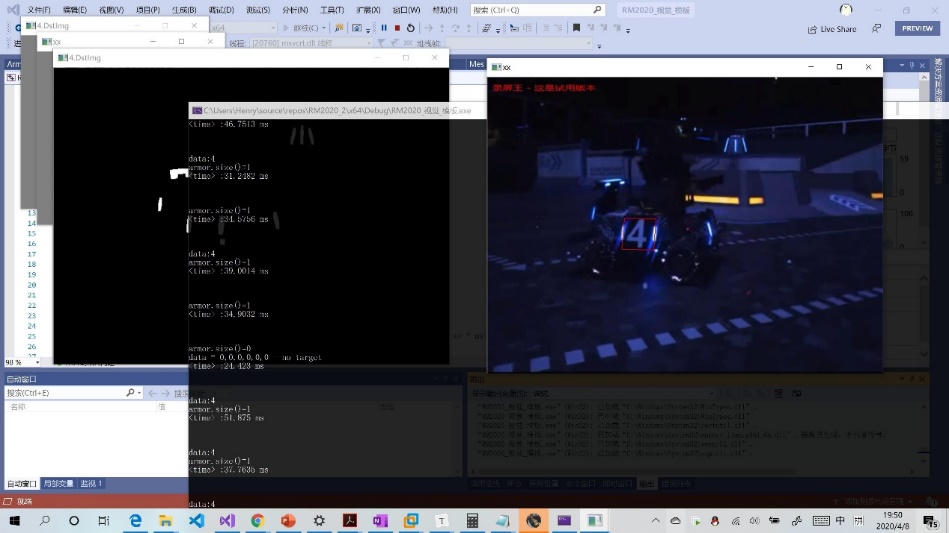
vector<float> descriptors;

流程：

1. 使用depart\_img，通过透射变换将目标装甲转换为正常rect图像（40x40）.
2. 使用hog提取特征。
3. 转换为SVM所需的单列图像后，进行匹配。
4. 返回匹配结果。

**方式二：欧氏距离识别：**

1. 使用adaptiveThreshold()函数，根据图像不同区域亮度分布，计算出领域的高斯阈值
2. 使用morphologyEx()函数，调用MORPH\_OPEN，对图片进行size(3, 3)的开运算处理。
3. 计算处理图的平均值，若小于阈值，则说明是负样本，返回0；若大于阈值，则：
   1. 使用morphologyEx()函数，调用MORPH\_OPEN，对图片进行size(2, 2)的开运算处理。
   2. 使用resize()函数将图片高度调整为和模板一致。
   3. 使用copyMakeBorder()函数将图片宽度填补为为和模板宽度一致。
   4. 使用norm()函数分别计算待测图片与模板的欧氏距离，判定距离最小的模板所对应的数字为检测出的数字结果。
4. 返回匹配结果。



识别效果见控制台输出

1. 存在的若干问题与思考

可参考github上的[链接](https://github.com/tsagaanbar/muc-rm-2020/blob/master/docs/intro.md#三有待优化处)。