



**摄像头数据采集**

1. 打开一个文本文件 "config.txt"，读取其中的配置信息。
2. 从文件中读取字符，将其存储在字符数组 aux\_str 中。
3. 根据特定字符的位置计算摄像头和串口的配置参数。
4. 打开指定摄像头和串口。
5. 设置摄像头的分辨率和帧率。
6. 创建一个窗口来显示画面。
7. 进入一个无限循环，处理每一帧的图像。
8. 从摄像头获取一帧图像，并进行高斯滤波和颜色空间转换。
9. 根据设定的阈值对图像进行二值化，以分离出冰球和机器人的轮廓。

**冰球轨迹追踪**

1. trackObjectPuck() trackObjectRobot() 跟踪冰球和机器人的位置。

* 初始化中心坐标
* 划定并框出感兴趣区域
* 找到符合hsv阈值的区域
* 遍历所有符合要求的区域，根据像素面积进行筛选
* 计算筛选后的区域的中心点坐标，保存在puckCenter中
* 把puckCenter坐标从摄像头坐标系转换为机器人坐标系
* 如果是机器人，程序结束；如果是冰球，继续执行以下语句
* 更新冰球当前XY坐标，绘制蓝色圆圈表示冰球当前位置
* 读取上一时刻的XY，画出冰球运动的轨迹

**冰球轨迹预测**

1. cameraProcess(33.33) 进行冰球轨迹预测。

* 输入参数time，表示两帧图像之间的时间间隔。
* 计算冰球在两帧图像中的x和y方向的位移差，存储在 vectorX 和 vectorY 变量中。
* 将当前的冰球速度赋值给上一次的速度变量 puckOldSpeedX 和 puckOldSpeedY。
* 位移差除以时间间隔，更新冰球的速度值 puckSpeedX 和 puckSpeedY
* 将平均速度值赋值给 puckSpeedXAverage 和 puckSpeedYAverage。
* 将 predict\_x\_attack 初始化为 -1。
* 如果冰球的y方向速度足够大，表示冰球纵向移动，进入追踪逻辑。
* 计算冰球的轨迹斜率 slope。
* 用GOTURN算法计算冰球最终目标点的坐标 predict\_x 和 predict\_y，通过将冰球的当前坐标和斜率带入公式得到。
* 如果预测的最终目标点超出了侧边界，表示冰球与侧边发生碰撞。
* 计算冰球与侧边界碰撞点的坐标 bounce\_x 和 bounce\_y。
* 计算冰球与侧边界碰撞后的预测时间 predict\_time。
* 根据碰撞后的斜率、碰撞点和预测目标点的y坐标，重新计算预测目标点的坐标。
* 如果预测的目标点仍然超出侧边界，表示冰球与另一侧边界发生了第二次碰撞，此时不进行新的预测，因为反弹两次后一般来说速度很低，没什么威胁
* 如果预测的目标点在侧边界范围内，计算最终预测目标点的坐标，并更新预测时间。
* 在图像上绘制冰球轨迹的线段。
* 如果冰球速度较慢或向另一边移动，将预测状态和相关变量重置为初始状态。

**冰球游戏策略（衔接运动控制系统）**

1. newDataStrategy() 根据当前状态给机器人制定一个策略。

* 将机器人状态设为回到初始位置（robot\_status = 0）。
* 如果冰球预测状态为1（冰球直冲过来），判断冰球预测的x坐标和时间，确定机器人的状态：
* 如果冰球预测的x坐标在151和323之间，并且预测时间小于300ms，将机器人状态设置为被动进攻模式（robot\_status = 2）。
* 如果冰球预测的x坐标在151和323之间，将机器人状态设置为防御模式（robot\_status = 1）。
* 如果以上条件均不满足，将机器人的x坐标限制在111和363之间，并将机器人状态设置为防御模式（robot\_status = 1）。
* 如果冰球预测状态为2（冰球碰撞到了墙壁），将机器人的x坐标限制在101和373之间，并将机器人状态设置为防御模式（robot\_status = 1）。
* 如果冰球预测状态为0，并且冰球的y坐标小于180（在机器人区域缓慢移动），则进行主动进攻：
* 如果攻击时间为0，则预测冰球400ms后的位置，并判断该位置是否在一定区域内。如果在区域内，设置攻击时间为当前时间加上400ms，并将机器人位置设置到冰球后方，同时将攻击状态设置为1。如果不在区域内，继续等待。
* 如果攻击时间不为0，根据攻击状态进行相应的操作：如果攻击状态为1（已经在冰球后方，准备进攻），并且攻击剩余时间小于200ms，进行进攻动作，并将攻击状态设置为2（进攻完成）。如果攻击状态为1，但攻击剩余时间大于200ms，将机器人位置设置到准备进攻区域。如果攻击状态为2（进攻完成），并且当前时间超过攻击时间加上150ms，重置攻击时间和机器人状态，并将攻击状态设置为0。
* 最后，将机器人状态设置为主动进攻模式（robot\_status = 3）。

1. 通过串口发送控制指令给机器人。
2. 计算帧率fps值并在图像上显示。
3. 显示处理后的图像。
4. 等待用户按键，如果按下了 ESC 键，则退出循环。