目　录

一 项目概述

一、项目概述

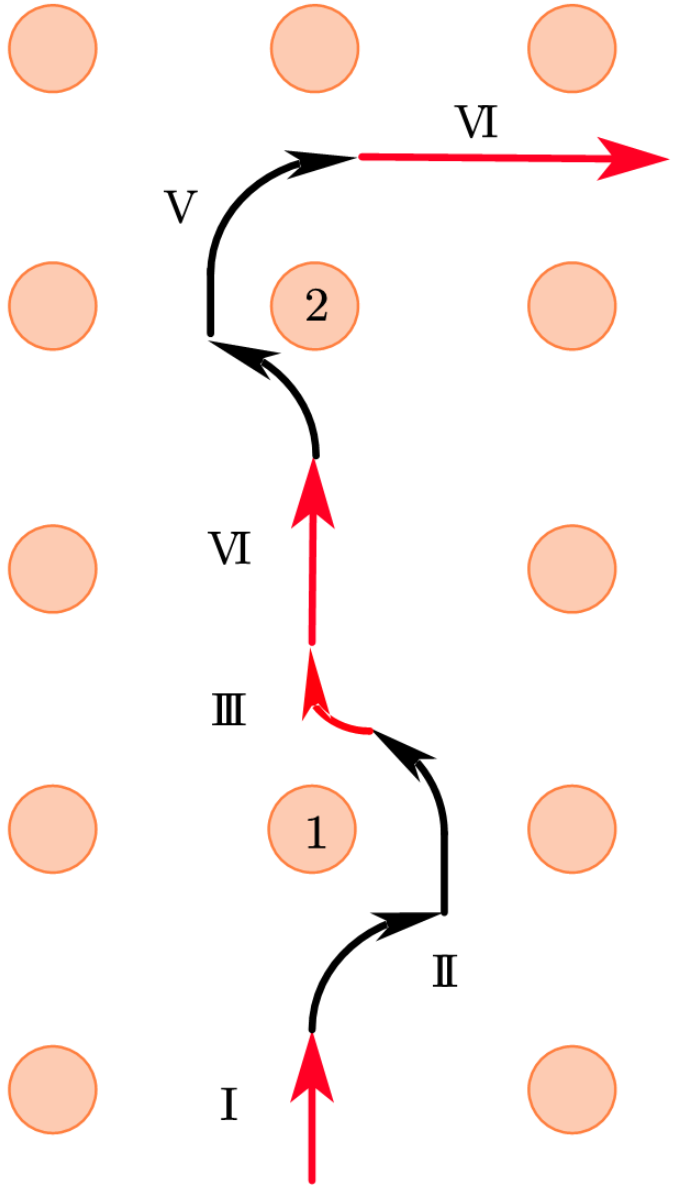
- 项目目标：实现机器人视觉控制，集成锥桶道路行驶与数字跟踪功能
- 实现了基于HSV颜色分割的锥桶检测算法
- 设计了五状态有限状态机，实现了完整的避障逻辑
- 实现了双ROI的闭环直线行驶算法
- 实现了基于模板匹配的数字识别算法
- 实现了基于位置和大小的双闭环控制策略

二 算法设计

二、算法设计：锥桶道路行驶

图像预处理：

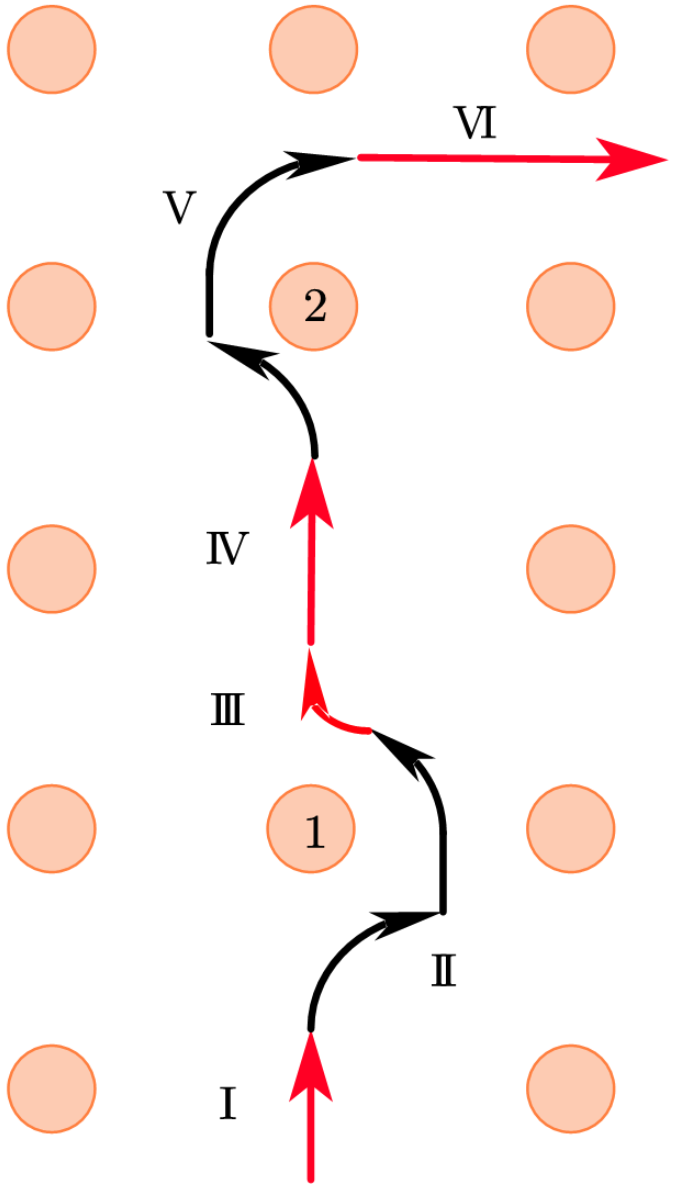
1. 高斯滤波：使用3 × 3 高斯核对图像进行平滑处理，减少噪声干扰。
2. 颜色空间转换：将图像从BGR 转换到HSV 颜色空间，便于颜色分割。
3. 颜色阈值分割：根据预设的HSV 阈值提取锥桶区域。
4. 形态学操作：先开运算后闭运算，去除噪点并填充空洞。



二、算法设计：锥桶道路行驶

锥桶检测（Ⅰ、Ⅲ、Ⅳ、Ⅵ）：

1. ROI 设置：在图像中下部（y=330）设置200×120像素的矩形区域作为检测区。
2. 轮廓提取：在ROI内使用findContours函数提取二值图像中的轮廓。
3. 中心点计算：通过计算轮廓的矩得到锥桶中心坐标。
4. 像素统计：统计ROI内锥桶颜色像素数量，用于障碍物判定。



二、算法设计：锥桶道路行驶

避障状态机设计：

状态0，对应Ⅰ（正常行驶，寻找锥桶1）：

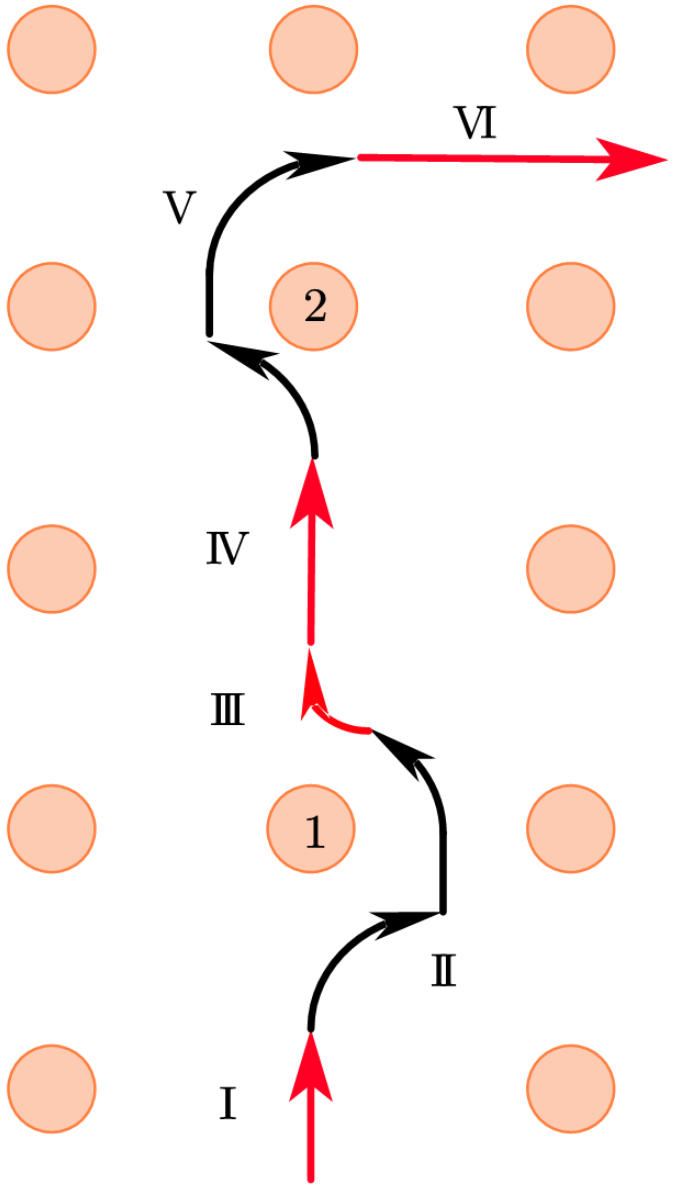
检测ROI内锥桶像素数量是否超过阈值，连续5帧检测到障碍物则进入状态1。

状态1，对应Ⅱ和Ⅲ（右行绕过锥桶1）：

Ⅱ部分为开环控制，分为右转-直行-左转三部分。

Ⅲ部分【提高稳定性】为根据锥桶位置进行对准调整，根据ROI中心与锥桶2中心间距进行比例调整。

```
float turn_gain = 0.05;
cmd.angular.z = (roi_center_x - cone_center.x) * turn_gain;
```



二、算法设计：锥桶道路行驶

避障状态机设计：

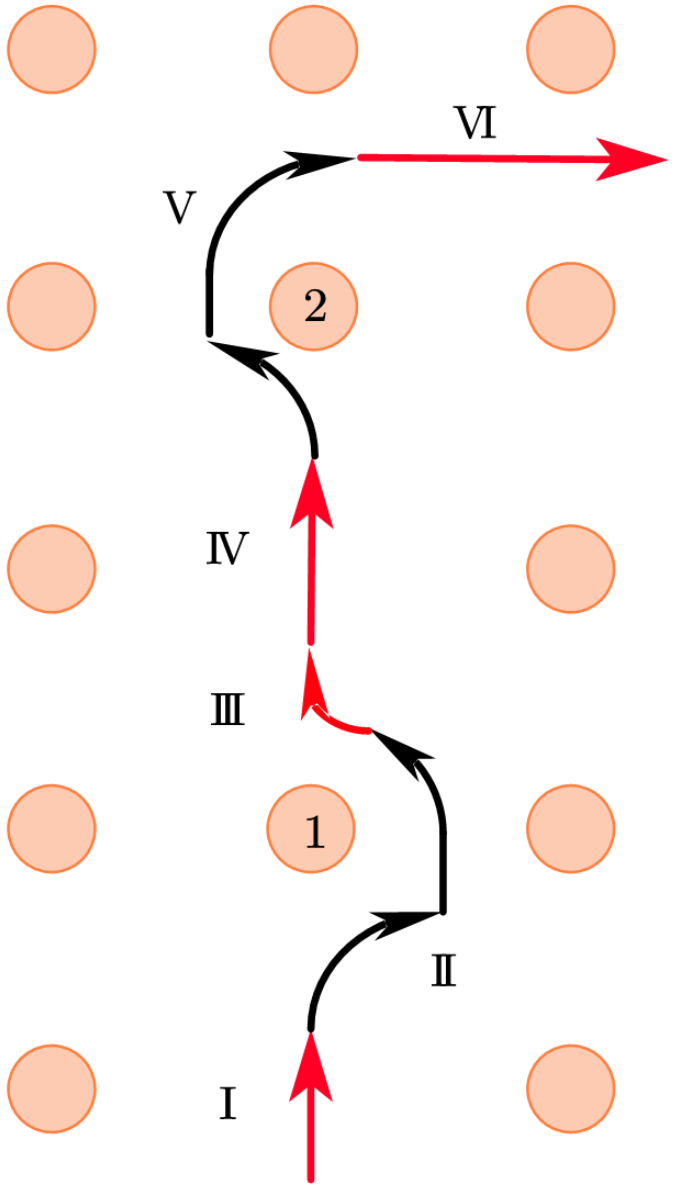
状态2，对应IV（寻找锥桶2）：逻辑同状态0。

状态3，对应Ⅴ（左行绕过锥桶2）：

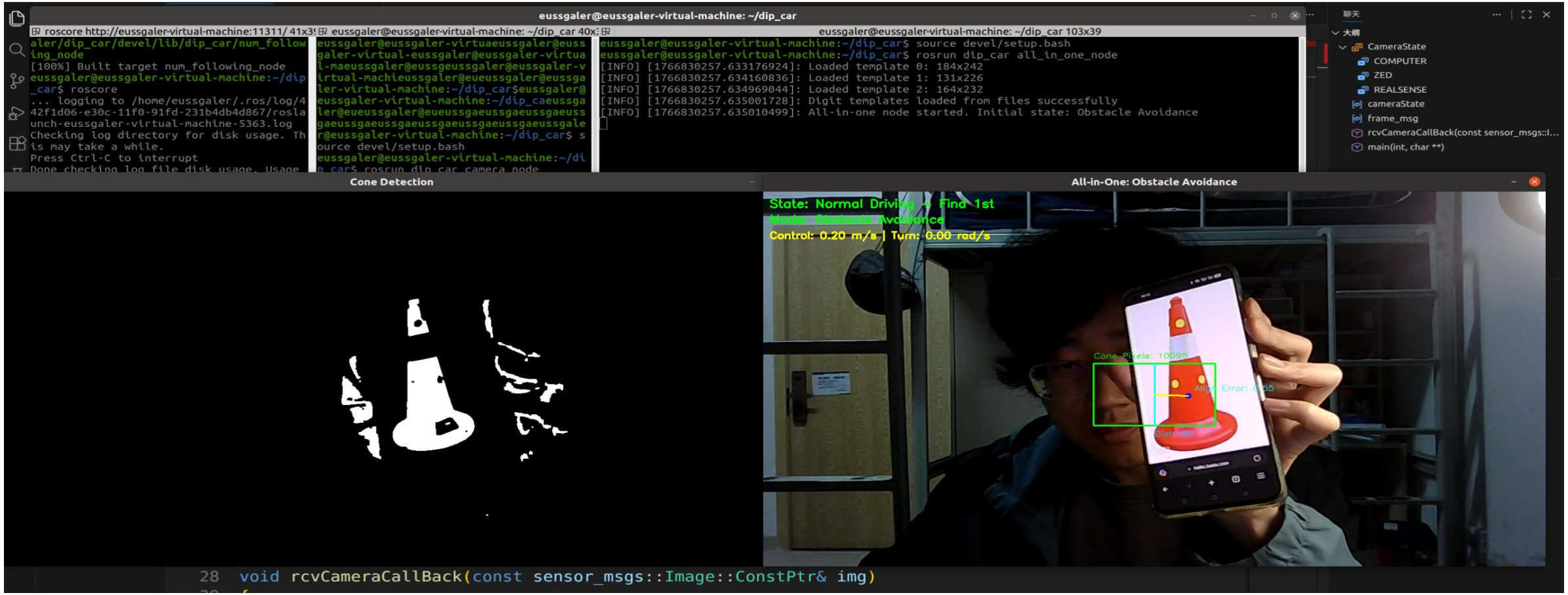
控制逻辑与状态1中的开环部分镜像对称。微调了参数使得能够直接进入状态VI直行阶段。

状态4，对应VI（道路保持阶段）：

在左右两侧设置ROI，分别检测左右锥桶。根据检测情况计算道路中心线：两侧均检测到则取中点作为道路中心；仅一侧检测到则保持固定安全距离；均未检测到则维持当前方向。



二、算法设计：锥桶道路行驶——避障

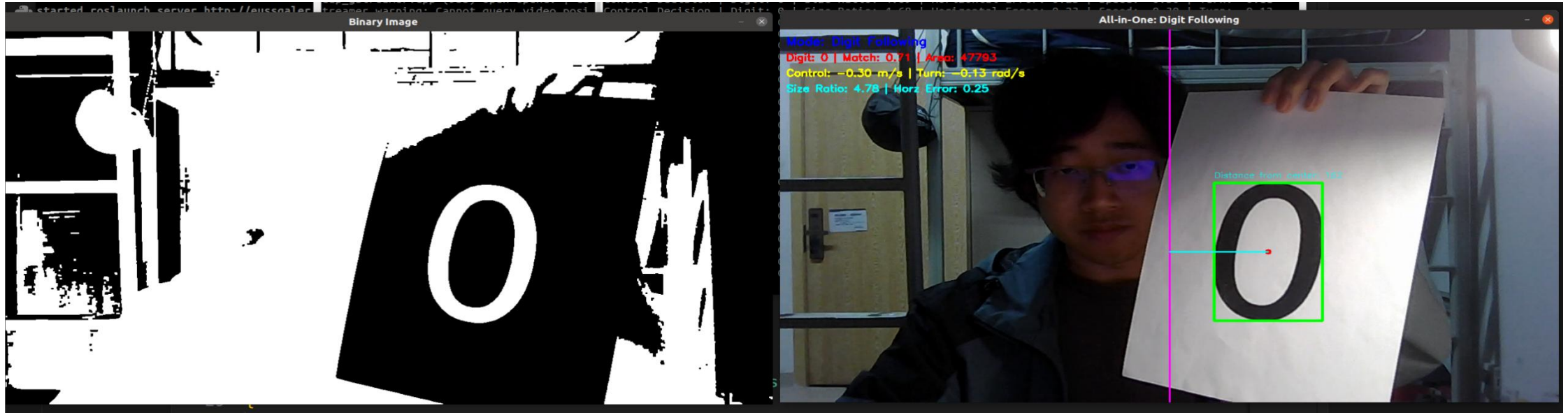


图像预处理与锥桶识别示意图

二、算法设计：数字跟踪

图像预处理：（状态VI直线行驶约30s后自动切换为数字跟踪模式）

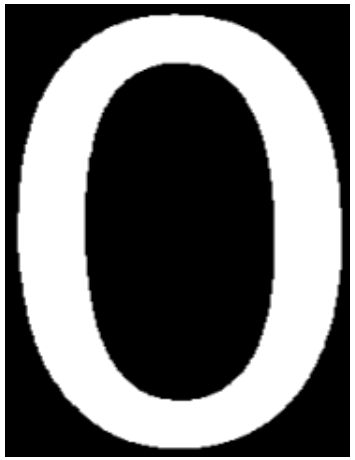
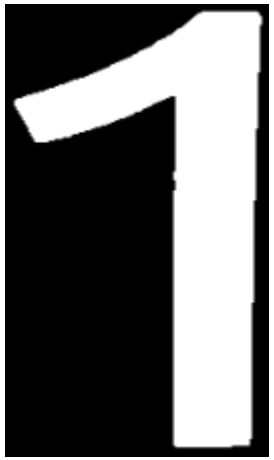
1. 灰度化：将彩色图像转换为灰度图像。
2. 直方图均衡化：增强图像对比度，提高数字区域识别率。
3. 二值化：采用OTSU自适应阈值法，自动确定最优阈值。
4. 形态学去噪：通过闭运算和开运算去除噪声点。



二、算法设计：数字跟踪

数字识别采用基于轮廓提取与多尺度模板匹配的方法：

1. 轮廓提取与筛选：筛选条件为轮廓面积在500-50000像素之间。
2. 模板加载：从文件加载数字模板。
3. 多尺度模板匹配：对候选区域先进行80%-120% 的尺度变换，令其更贴合模板的尺寸。
4. 匹配决策：选择匹配分数最高且超过阈值的模板作为识别结果。


二、算法设计：数字跟踪

数字追踪采用基于位置与大小的双闭环控制策略：

1. 距离控制：根据数字区域面积与目标面积的比值调节线速度。

```
float sizeRatio = currentSize / g_targetArea;
if (sizeRatio > (1.0f + AREA_TOLERANCE)) {
   cmd.linear.x = -MAX_SPEED * min(1.0f, sizeRatio - 1.0f);
} else if (sizeRatio < (1.0f - AREA_TOLERANCE)) {
   cmd.linear.x = MAX_SPEED * min(1.0f, 1.0f - sizeRatio);
}
```

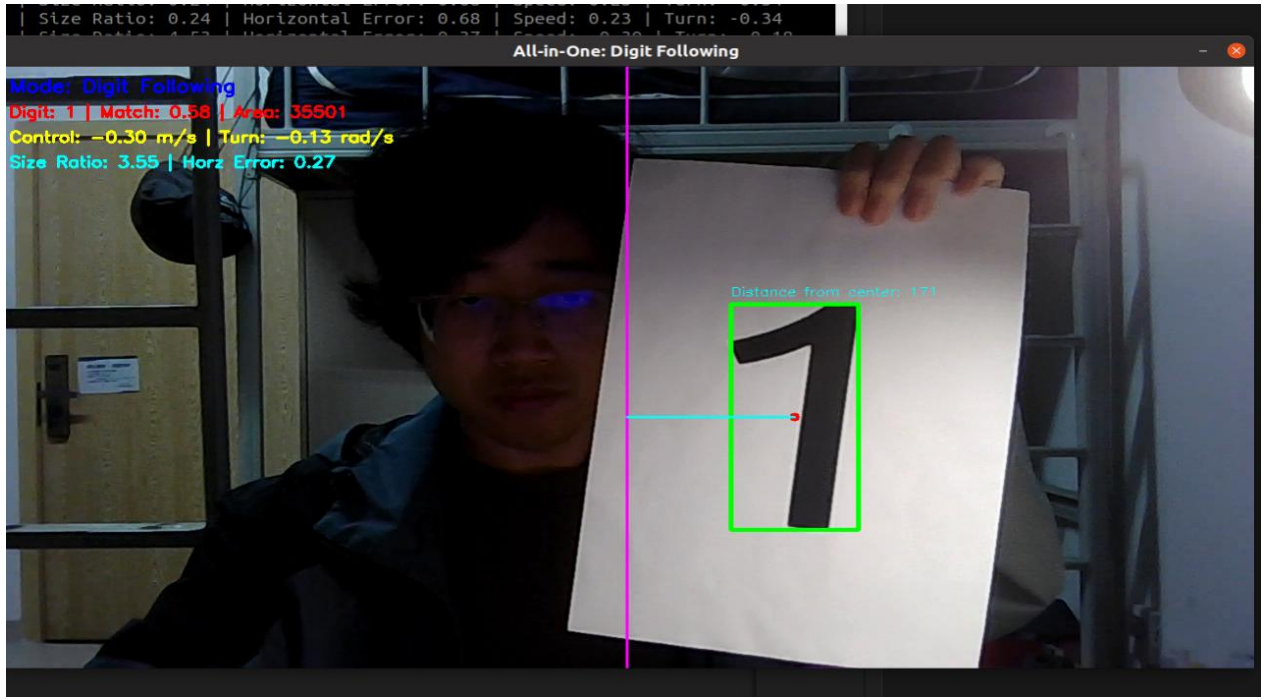
2. 方向控制：根据数字中心与图像中心的水平偏差调节角速度。

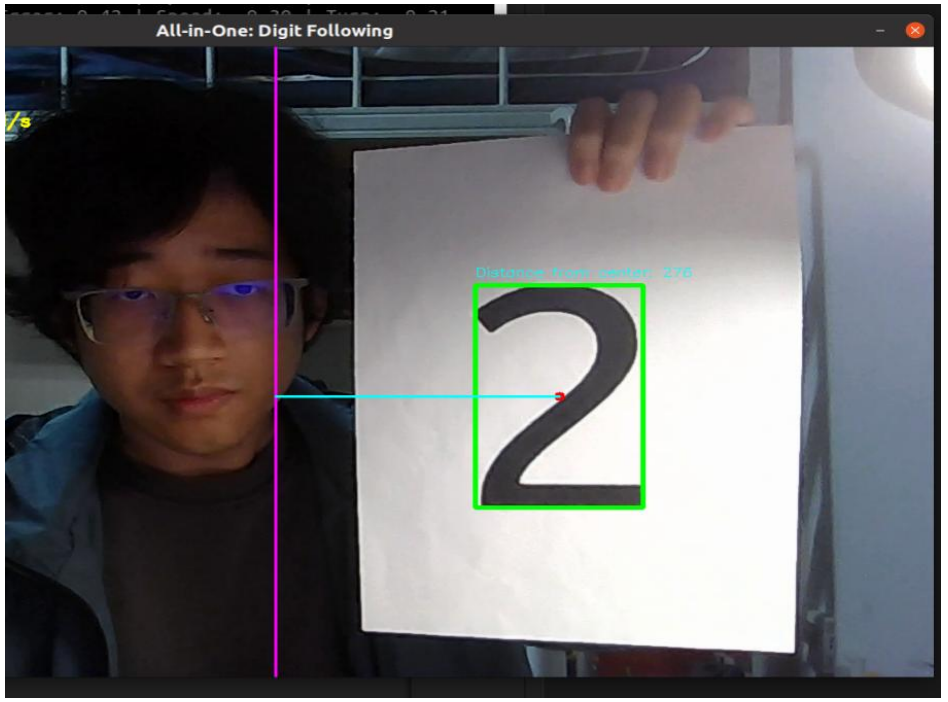
```
float horizontalError = (digitCenter.x - src.cols/2) / (float)(src.cols/2);
cmd.angular.z = -horizontalError * 0.5;
```

二、算法设计：数字跟踪

3. 容差机制：面积容差设为15%，在此范围内不调整距离。

4. 丢失帧处理：连续10 帧未检测到数字则停止运动。





三 收获体会

三、收获体会

通过本次课程设计：

- 深入理解了数字图像处理的基本原理和方法；
- 掌握了使用OpenCV 进行图像处理的编程技能；
- 提高了C++ 编程能力和软件工程实践能力；
- 培养了系统集成和调试解决问题的能力；

解决的主要问题：

- 全开环方法不稳定：引入适量的闭环调节，平衡算法复杂度和调试难易度；
- 数字识别准确率：选择合适的二值化方法、选择合适大小的模板；

请批评指正