精准灭火机器人目标定位系统

项目概述

本项目旨在通过红外热成像视觉技术，实现对火源的智能识别、定位与分组，为后续的精准灭火机器人提供精确的目标喷射坐标。系统从红外相机获取温度数据，通过图像处理算法检测高温热点，并根据预设规则（如热点间的距离）对相邻热点进行分组，最终输出每个火灾区域的中心坐标。

主要功能

- **热点检测：** 从红外相机提供的温度矩阵中，准确识别超出预设温度阈值的区域作为潜在火源。
- **噪声过滤：** 通过形态学处理和面积阈值过滤，去除图像噪声和过小的无效热点。
- **热点分组：** 根据热点之间的近似真实世界距离（基于相机标定和场景假设），将物理上邻近的多个热点合并为一个火灾区域。
- **目标定位：** 为每个独立的火灾区域或单个热点计算一个中心喷射坐标（像素坐标及估算的近似世界坐标）。
- **优先级输出：** （可选）根据火灾区域的严重性（如面积、温度等综合评估）对目标进行排序输出。
- **可视化：** 提供处理过程和结果的可视化界面，方便调试和演示。

技术栈

- **编程语言：** C++
- **核心库：** OpenCV (用于图像处理、矩阵运算、轮廓检测等)
- **开发环境：** VS Code, CMake, g++
- **依赖硬件：**
  - 红外热成像相机 (需提供C++ SDK以获取温度矩阵)
  - 工控机，配置：
    - CPU： i5-5200U@2.2GHz
    - 运行内存： 8GDDR3L
    - 硬盘： 64GB

系统流程

1. **图像采集：** 通过红外相机SDK获取实时的温度数据矩阵 (通常为 `float` 类型)。
2. **预处理：**
   - （可选）图像去畸变（如果相机畸变严重且已标定）。
   - （可选）噪声平滑。
3. **热点分割：**
   - 应用温度阈值，将温度矩阵转换为二值图像，高亮潜在火源区域。
   - 进行形态学操作（如开运算、闭运算）以优化二值图像，去除噪声，连接断裂区域。
4. **轮廓提取与筛选：**
   - 使用 `cv::findContours` 提取所有独立高温区域的轮廓。
   - 根据面积、形状等因素过滤掉无效轮廓。
5. **特征计算与坐标转换：**
   - 计算每个有效热点的质心（像素坐标）、最高温度、面积等特征。

    - 利用预先标定的相机内参和场景假设（如火源平面距离），将热点质心的像素坐标近似转换为世界坐标。
6. **热点分组：**
    - 遍历所有有效热点。
    - 如果两个或多个热点的近似世界坐标距离小于预设阈值（例如1米），则将它们视为一个火灾群组。
7. **喷射目标确定：**
    - 对于每个火灾群组，计算其内部所有热点质心的平均值，作为该群组的最终喷射目标坐标。
    - 对于未被分组的独立热点，其自身质心即为喷射目标坐标。
8. **结果输出：**
    - 输出计算得到的每个喷射目标的像素坐标和（估算的）世界坐标。
    - （可选）按严重性对目标进行排序。

安装与配置

依赖项

- **OpenCV:** 版本 4.11.0 。确保已正确安装并配置到您的开发环境中。
    - 安装指南: OpenCV官方安装文档链接
- **红外相机SDK:** (相机型号及其SDK名称和版本)。
    - 请遵循相机制造商提供的SDK安装和配置指南。

相机标定

为了实现基于真实世界距离的热点分组和相对准确的世界坐标估算，必须对红外相机进行精确标定。

1. **准备标定板：** 制作或购买适用于红外相机的标定板（可能需要考虑材质对红外辐射的反射/吸收特性）。
2. **采集标定图像：** 从不同角度和距离拍摄标定板的红外图像。
3. **运行标定程序：** 使用OpenCV的相机标定函数（如 `cv::calibrateCamera`）或第三方标定工具，计算相机的内参矩阵 (`cameraMatrix`) 和畸变系数 (`distCoeffs`)。
4. **配置参数：** 将标定得到的 `cameraMatrix` 和 `distCoeffs` 更新到项目代码中的相应位置 (例如，`main.cpp` 或配置文件中)。
5. **场景假设参数：** 根据应用场景，合理设置 `ASSUMED_DISTANCE_TO_FIRE_PLANE_METERS` （假设火源平面距离）等参数。如果无法做此假设，基于世界距离的分组逻辑需要调整。

代码结构

```
├── src/                    # 源代码目录
│   ├── main.cpp            # 主程序入口
│   ├── visionModule.h      # 视觉处理函数声明
│   ├── visionModule.cpp    # 视觉处理函数实现
│   └── utils.h             # 数据结构定义 (HotSpot, SprayTarget)
├── include/                # 存放项目内部头文件，如相机SDK头文件
├── CMakeLists.txt          # CMake 编译配置文件
├── README.md               # 本文件
└── config/                 # 存放配置文件
    └── camera_params.xml   # 存放相机标定参数
```