基于PyQt5的分布式移动端目标检测监控平台

软件需求规约

版本 <1.0>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| <12/11/2024> | <1.0> | <初始内容> | <徐郅浩、米博宇、郑心锐、李盼奇> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目录

1. 简介 4

1.1 目的 4

1.2 定义、首字母缩写词和缩略语 4

1.3 参考资料 4

2. 整体说明 5

3. 功能需求 5

3.1 <Use case 图> 6

3.2 <Use case1 规约> 6

3.3 <Use case2 规约> 6

4. 非功能需求 7

4.1 易用性 7

4.2 可靠性 7

4.3 性能 7

4.4 可支持性 8

4.5 设计约束 8

5. 其它产品需求 8

5.1 联机用户文档和联机帮助的需求 8

5.2 接口需求 8

5.2.1 用户界面 8

5.2.2 硬件接口 8

5.2.3 软件接口 9

5.2.4 通信接口 9

5.3 适用的标准 10

软件需求规约 (简化版)

# 简介

## 目的

本文档为“基于PyQt5的分布式移动端目标检测监控平台”项目的软件需求规约文档，对软件做了整体的说明，并从用户角度提供了主要功能的use case图与具体事件流描述，使得用户能更好地认识软件的功能。同时，文档对软件的非功能需求，如性能、可靠性等做了充分的定义和介绍，明确了本软件需要达到的完成度。另外，本文档还介绍了软硬件接口方面的内容，便于用户集成差异化的模型，或基于定制化硬件进行扩展。

## 定义、首字母缩写词和缩略语

本文档所需的所有术语定义，缩写等信息如表所示：

|  |  |  |  |
| --- | --- | --- | --- |
| 术语中文名称 | 术语英文名称 | 术语缩写 | 术语含义 |
| 目标检测 | Object Detection | OD | AI任务场景，检测图像中的目标位置与类型 |
| 目标计数 | Object Counting | OC | AI任务场景，统计图像中各类型目标的数目 |
| 异常检测 | Anomaly Detection | AD | AI任务场景，检测实时图像中是否出现异常类型物体 |
| 实时流协议 | Real Time Streaming Protocol | RTSP | 常用应用层协议，控制实时数据的传送 |
| 吞吐量 | Throughput | Throughput | 单台服务器上，模型能够在一秒内处理的AI推理任务数目 |
| 每类平均精度 | Mean Average Precision | mAP | 目标检测等场景常用评价指标 |
| 召回率 | Recall Rate | Recall | 目标检测等场景常用评价指标 |
| 每秒帧数目 | Frame Per Second | FPS | 视频文件基本参数 |
| 掉帧率 | Drop Rate | Drop Rate | 一段时间内为保证视频实时性，主动跳过的帧占总帧数的比重 |
| Python Qt | Python Qt | PyQt | 项目主要技术框架，基于Python前端的Qt开发平台 |
| 回调函数 | Callback Function | Callback Func | 在PyQt编程中，某个事件触发后，引擎会主动进行异步调用的函数 |
| 面向事件编程 | Event-Oriented Programming | EO Programming | PyQt基本编程范式 |

## 参考资料

1. YOLOv5模型：<https://github.com/ultralytics/yolov5>

2. OpenCV-Python文档：<https://docs.opencv.org/3.4/df/d9d/tutorial_py_colorspaces.html>

3. PyQt5文档：<https://doc.qt.io/qtforpython-6>

# 整体说明

1.产品总体效果

该产品定位于家庭、教室和室内办公环境的智能监控系统，集成易用且稳定的智能监控解决方案，通过用户友好的界面和智能监控功能，提升用户体验，满足用户对安全和便捷的需求，提高生活和工作的安全性与效率。

2.产品功能

包括实时监控、目标检测、目标跟踪、目标计数和异常检测。

3.用户特征

目标用户群体：家庭用户、企业用户以及需要监控解决方案的机构。

用户技能水平：用户无需专业技术背景，通过简单的指导即可安装和操作系统。

用户期望：用户期望系统能够提供远程监控、证据保存和智能化管理。

4.约束

时间限制：项目有明确的计划进度和迭代开发时间表。

预算限制：项目需要考虑开发成本和硬件成本，以确保产品具有成本效益。

技术限制：需要考虑深度学习模型的精度和响应速度，以及多平台兼容性问题。

法律和合规性：项目需确保数据传输和存储符合隐私保护法规。

5.假设与依赖关系

市场假设：随着智能家居和办公自动化的快速发展，视频监控系统的应用愈发广泛。

技术假设：基于PyQt5的开发框架能够提供跨平台支持和易于扩展的特性。

依赖关系：项目依赖于Python和PyQt5的开发环境，以及深度学习算法的成熟度。

6.需求子集

功能性需求：系统需要实现多摄像头接入、实时数据处理和智能分析。

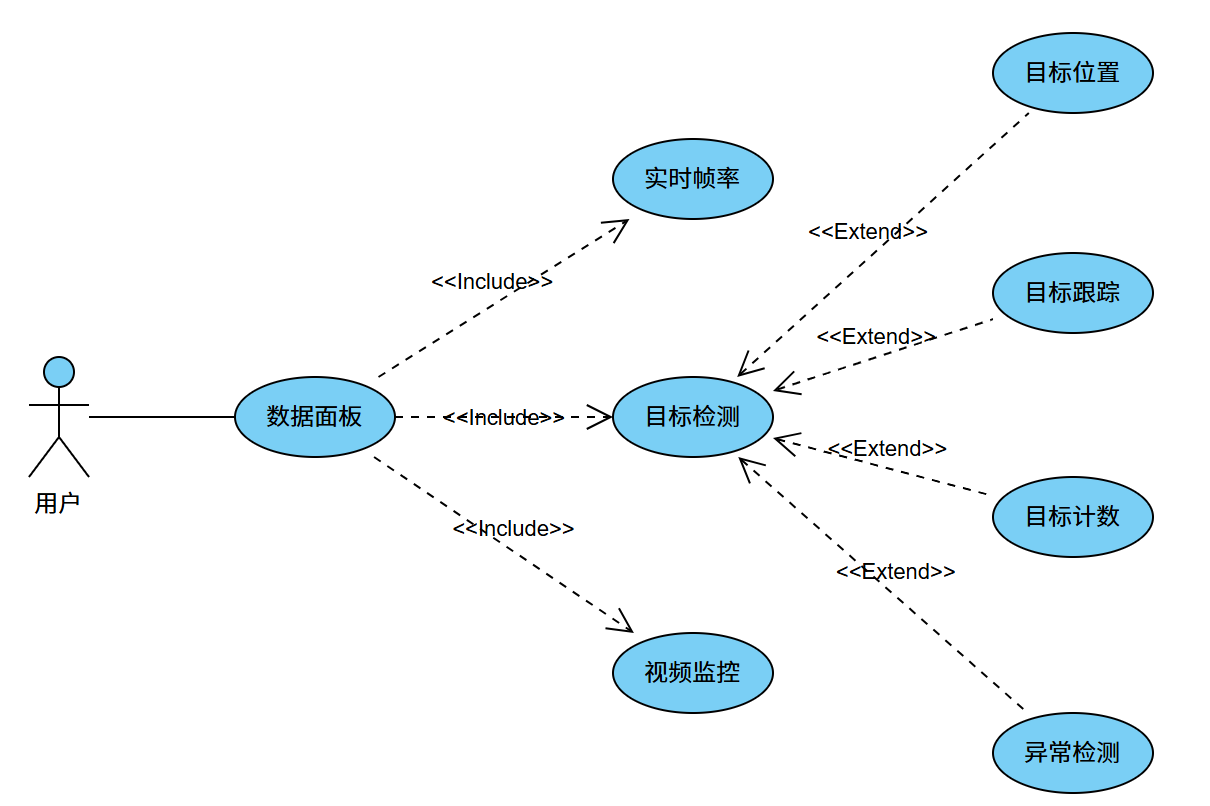
非功能性需求：包括系统的稳定性、安全性、响应速度和用户界面的易用性。

合规性需求：系统必须遵守数据隐私和安全相关的法律法规。

用户故事和用例：用户可以通过手机或电脑远程查看监控情况，系统自动对场景中的目标进行检测。

# 功能需求

## <Use case 图>



数据面板：用户与系统交互的界面，展示了实时处理的分析结果

实时帧率：展示视频帧的处理速度，确保视频流的平滑性和及时性

视频监控：将处理过的实时视频流在用户界面上呈现

目标检测：对视频帧进行物体识别，实时检测监控画面中的目标物体

目标位置：通过图形化方式展示检测到的目标物体在视频中的位置

目标跟踪：在目标检测的基础上，系统支持对检测到的目标进行持续追踪

目标计数：持对某一时间段内进入或离开监控画面中的目标进行计数

异常检测：通过分析正常环境下的目标活动模式，系统能够识别异常行为或事件

## <Use case1 规约>

用例名称：数据面板

参与者：用户

用例描述：用户与系统交互的界面，展示了实时处理的分析结果

触发器：用户打开软件

前置条件：用户安装了软件

基本事件流：

1. 用户打开数据面板
2. 系统接收来自传感器的原始视频流，并在此基础上进行实时的目标检测处理
3. 将实时处理的分析结果展示在数据面板上

扩展事件流：

如果没有传感器或传感器失效，系统无法接收视频流，数据面板上没有实时结果。

## <Use case2 规约>

用例名称：目标检测

参与者：用户

用例描述：对视频帧进行物体识别，实时检测监控画面中的目标物体

触发器：用户打开目标检测功能

前置条件：数据面板被用户打开

基本事件流：

1. 用户打开目标检测功能
2. 系统将从传感器获取的连续视频帧数据输入到目标检测模型中，目标检测模型利用深度学习算法，实时分析视频帧中的关键目标物体，并输出检测结果
3. 将实时处理的检测结果展示在数据面板上

扩展事件流：

如果没有传感器或传感器失效，系统无法接收视频流，目标检测模型没有输入，无法输出检测结果。

# 非功能需求

## 易用性

 **用户培训**：普通用户通过30分钟的基础培训可以掌握系统的主要功能；高级用户可以在2小时内掌握所有摄像头配置、目标检测模型部署、数据采集等高级功能。

 **易用性标准**：系统遵循常见的GUI设计标准，界面布局直观、操作流程简便，符合Microsoft的GUI标准和其他常见界面交互规范，确保用户能够流畅地执行基本操作。

 **任务效率**：用户可在平台内快速完成对多个摄像头的切换、数据采集等操作，系统设计优先考虑减少鼠标点击次数，提升操作便捷性。

## 可靠性

 **可用性**：系统设计目标为全天候可用，即达到99.9%的可用时间，允许日常维护和系统升级期间安排停机。

 **平均故障间隔时间 (MTBF)**：系统应在连续运行200小时后才允许发生一次系统性故障，确保平台在监控过程中保持稳定。

 **平均修复时间 (MTTR)**：在发生非硬件性故障时，系统的恢复时间不应超过5分钟。

 **精确度**：系统的视频显示和数据监控模块应符合摄像头的帧率精度要求，确保显示的实时帧率和掉帧率与实际情况保持误差不超过0.1%。

 **缺陷率**：系统代码缺陷率不超过0.5个错误/千行代码 (bugs/KLOC)；对视频采集和数据存储等核心功能模块的重大缺陷要求降至0.1个错误/千行代码。

## 性能

 **响应时间**：摄像头图像实时预览的平均响应时间应小于2000毫秒，最大响应时间不超过5000毫秒。

 **容量**：系统支持最多4台摄像头的数据接入，能够满足未来扩展需求。

 **资源利用**：在默认配置下，系统单个摄像头的处理进程占用的CPU不超过25%，总内存使用率不超过4GB，确保在多任务并发时的资源使用稳定性。

## 可支持性

 **编码标准**：系统采用Python编码规范，并遵循PEP8编码标准；关键模块的代码有详细注释，便于后期维护。

 **命名约定**：所有变量、函数、类命名遵循清晰的命名规则，以提高代码的可读性和可维护性。

 **库和框架**：使用Python-QT作为可视化界面框架、opencv-python进行视频流处理、torch框架搭建目标检测模型，确保系统的扩展性。

 **维护访问**：系统提供详细的日志记录功能，便于后期调试与维护，维护人员可通过管理界面和日志文件快速定位和解决问题。

## 设计约束

 **开发语言**：系统主要使用Python语言开发。

 **开发工具**：系统推荐使用PyCharm或VS Code进行开发，统一开发环境和调试工具，以保证开发一致性。

 **架构要求**：系统架构基于多进程并发设计，每个摄像头的显示界面独立运行在单独的进程中，实现拍摄与推理任务的解耦。

 **第三方库**：系统使用IP摄像头软件对手机端进行控制，并采用multiprocessing库进行并发处理，满足实时控制与多摄像头并行处理的要求。

# 其它产品需求

## 联机用户文档和联机帮助的需求

1. 用户手册：提供系统的详细使用手册，包括系统的功能描述、操作流程、常见问题解答内容
2. 帮助中心：提供系统的常见问题解答、操作指南等内容，以便用户快速定位并解决问题
3. 反馈机制：提供用户反馈机制，以便用户反馈问题或者提出建议，并通过预留联系方式获得及时的回复和处理。

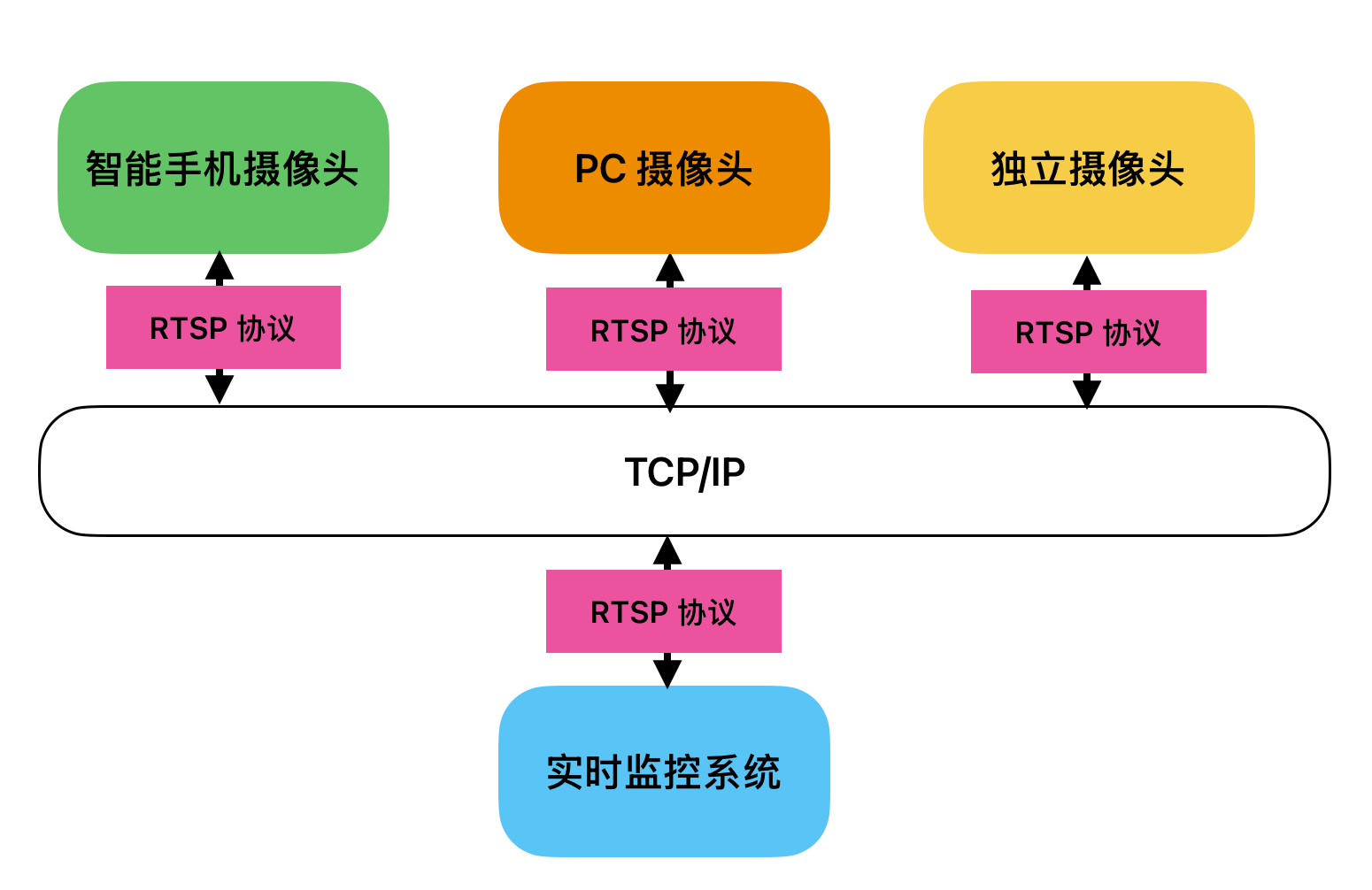
## 接口需求

该实时监控软件的接口需求包括用户界面、硬件接口、软件接口和通信接口等。系统将支持多种协议和端口，以确保数据的实时采集和展示。特定的逻辑地址将用于与设备和其他软件系统的兼容性

### 用户界面

* 菜单栏：为系统提供详细的设置选项，包括程序文件操作（File）、窗口设置（Windows）、视图调整（View）、其他选项（Setting）；
* 监视区：提供实时的视频流界面显示和选择。较大的窗口显示实时的视频流，其下方的按钮提供显示开关、选择视频流设备、控制视频流设备选项；
* 控制区：允许用户设置和调整不同设备的配置，控制视频记录和截图，依照不同设备的属性可能展示不同的可用选项；
* 实时数据区：展示视频流的系统数据，在接入深度学习自动化处理模块后显示视频流的处理结果（如异常警告和物体检测结果）；
* 

### 硬件接口



本系统需要与智能手机与PC等的摄像头通过网络适配器连接以太网或无线网络进行实时数据流通信，逻辑拓扑结构如上图所示，支持MAC地址动态分配。

### 软件接口

1. 机器学习模型接口：
   1. 模型类型：YOLOv3 等深度学习模型
   2. 协议：ONNX
   3. 功能：实时图像处理与物体检测

### 通信接口

1. 用户层通信接口：
   1. 协议：RTSP（实时流传输协议）。
   2. 功能：与手机或PC摄像头建立实时视频流连接，支持高分辨率视频传输和处理。
2. 局域网接口：
   1. 协议：TCP/IP
   2. 端口：554
   3. 逻辑地址：支持 IPv4

## 适用的标准

1. 质量标准：ISO 9001：确保软件开发和管理的质量。
2. 通信标准：RTSP（实时流传输协议）和TCP/IP协议。
3. 数据安全标准：ISO/IEC 27001：确保信息安全管理。
4. 用户界面标准：ISO 9241：确保软件的可用性和人机交互设计。