西南大學

本科毕业论文（设计）

题　　目 基于无人机图传的图像目标低延时检测系统设计

学　　院 计算机与信息科学学院 软件学院

专　　业 自动化（中外合作办学）

年　　级 2019 级

学　　号 222019321132096

姓　　名 施哲君

指 导 教 师 范子川

成　　绩

2023 年 5 月 24 日

目 录

基于无人机图传的图像目标低延时检测系统设计

施哲君

西南大学计算机与信息科学学院，重庆，400715

摘　要：随着无人机领域的探索逐渐深入，无人机的发展逐渐加快了步伐，无人机的潜力正逐渐被发掘。目前，无人机系统种类繁多、用途广泛特点鲜明。其中，无人机图传是无人机的重要视觉功能。本文设计了一种基于无人机图传的图像设计对选定目标的低延时检测系统，面向无人机操控人员提供辅助功能，拓展无人机的功能性，丰富无人机的应用场景。搭载该系统的无人机，可以广泛应用于航拍摄影、检修运维、抢险搜救、军事任务等场景。该系统主要由无人机图传系统和目标检测跟踪系统两大系统组成，针对无人机图传的图像特点，该系统致力于对图传图像的传输延时控制和高效可靠的目标检测功能优化。基于 Python 语言开发，以 OpenCV 和跟踪算法作为基础，不依赖 GPU 加速。最终整个系统可以部署至满足性能要求的便携式移动设备，便于无人机操作人员得到实时信息，及时进行反应。

关键词：无人机跟踪；判别式相关滤波器；通道和空间可靠性跟踪器

Design of Low Latency Detection System for Image Objects Based on UAV Graphic Transmission

SHI Zhejun

School of Computer and Information Science, Southwest University, Chongqing 400715, PR China

Abstract: With the gradual exploration of the UAV field, the development of UAVs has gradually accelerated, and the potential of UAVs is gradually being explored. At present, there are a wide range of UAS with distinctive features for a wide range of uses. Among them, UAV map transmission is an important visual function of UAVs. In this paper, we design a low latency detection system for selected targets based on UAV image design for UAV operators to provide auxiliary functions, expand the functionality of UAVs, and enrich the application scenarios of UAVs. UAVs equipped with this system can be widely used in aerial photography, maintenance and operation, rescue and search, military missions and other scenarios. The system mainly consists of two systems: UAV image transmission system and target detection and tracking system. For the image characteristics of UAV image transmission, the system is dedicated to the control of transmission delay of image transmission and optimization of efficient and reliable target detection function. Developed based on Python language, it is based on OpenCV and tracking algorithm and does not rely on GPU acceleration. Eventually the whole system can be deployed to portable mobile devices that meet the performance requirements, facilitating UAV operators to get real-time information and react in time.

Key words: UAV tracking; Discriminative Correlation Filter; Channel and Spatial Reliability Tracker

第 1 章　导论

1.1 引言

无人驾驶飞行器简称为无人机，它可以利用无线电遥控设备或自主的程序控制装置进行操作[1]。随着无人机领域的深入探索，无人机的发展步伐正在逐渐加快，其潜力正在不断被挖掘。目前，无人机系统的种类繁多，应用范围广泛，其特点在尺寸、重量、航程、飞行时间、飞行高度、飞行速度以及执行任务的方式等方面有很大的差异。航行监控是无人机工作流程中至关重要的一部分，其内容包括实时掌握无人机的姿态、方位、空速、位置、电池电量、即时风速和风向、任务时间等关键状态。这些信息有助于操作人员实时判断任务的可执行性，并进一步保证任务的安全性[2]。

无人机图传是航飞监控的重要组成部分。具体而言，无人机图传是指无人机通过搭载图传设备将拍摄的实时图像、视频传输到地面控制站或其他设备上，实现远程观察和控制。无人机图传是实现无人机自主飞行、目标检测、地图构建等功能的关键技术之一。无人机图传一般由无线图传模块和摄像头组成。无线图传模块负责将摄像头采集到的图像或视频信号通过无线信号传输到地面控制站，同时还要保证传输的稳定性和实时性。摄像头则需要具备高清晰度、高帧率、广角等特点，以满足无人机在不同场景下的实时拍摄需求。

无人机图传的应用广泛，将无人机图传和计算机视觉技术结合起来的应用，称之为无人机视觉。无人机视觉利用视觉传感器等设备获取环境信息，实现自主飞行、目标检测、地图构建等功能。随着计算机视觉和无人机技术的不断发展，无人机视觉应用逐渐扩展，取得了一系列新的进展。其中，目标检测技术是无人机视觉的核心之一，其目的是在无人机的视野范围内检测到目标物体。

本文主要研究的是基于无人机图传设计对用户选定目标的低延时检测系统，首先研究了基于无人机平台的图像传输特点，并确定了将图像数据传输至部署平台用于目标检测的方案。其次研究了目前主流的跟踪算法，基于各类算法的特点进行比较，评估优劣势和可行性之后确定了适用的算法，并基于该检测算法设计了满足要求的跟踪器。最后通过软件仿真和实机测试对比了经典跟踪器和所设计的系统的性能，验证了所设计的系统的高效性、可靠性，以及与无人机平台相结合的优势。

1.2 研究背景及意义

1.2.1 研究背景

本文设计的系统重点包括模拟图传和检测算法两个方面。模拟图传是一种传输图像信号的技术，相较于数字图传有着诸多先天的优势。首先，模拟图传的实时性强，无需等待数据压缩、解压缩等处理，可以实现实时传输，非常适合用于需要实时反馈的应用场景。其次，模拟信号的传输距离更远，部分型号的发射机/接收机可以达到数十公里的传输距离。此外，由于模拟信号是连续的波形信号，相较于数字信号，具有更强的抗干扰能力，使得信号传输的过程更为稳定。随着技术的不断发展，模拟图传也经历了几个重要的发展阶段，最初模拟图传使用基带调制技术，将图像信号直接传输到显示器上。这种技术具有简单、可靠的优点，但其传输距离较短、受到电磁干扰较大等缺点，限制了其在实际应用中的发展。之后的模拟图传采用了频率调制和振幅调制等技术，可以传输较远距离的图像信号。此外衍生出的诸如自动增益控制、多路复用等技术，造就了现在集成多种功能的高性能图像传输技术。

检测算法的实现，其核心在于图像跟踪技术。图像跟踪技术是指通过计算机算法实现对目标在图像序列中的位置、形状等属性的追踪。基于颜色的图像跟踪技术是最早被广泛使用的一种图像跟踪技术。该技术的原理是根据目标物体的颜色信息进行跟踪，相比于基于形状和纹理的跟踪方法，这种跟踪方法更加简单、快速，但同时也存在颜色分布不稳定、容易受到环境光线变化等问题。随之出现的是基于模版匹配的图像跟踪技术，他的特点是通过构建目标模版，计算目标模版与当前帧图像的相似度来实现目标追踪。这项技术相对于基于颜色的跟踪方法而言，在精度和鲁棒性方面都有了一定的提升，不足之处在于目标形变、旋转、尺度变换等情况下的跟踪效果不理想。之后发展出基于特征点图像跟踪技术，这是一种基于目标图像特征的跟踪方式，通过提取第一帧图像的特征点，计算这些特征点在下一帧图像中的位置，从而实现目标跟踪[3]。相较于先前的跟踪技术，它具有了一定的鲁棒性，但是检测过程中遇到遮挡、旋转等情况，容易出现跟踪失败的问题[4]。近年来较为流行的是基于深度学习的图像跟踪技术，这种技术主要通过利用深度神经网络从大量的图像数据中学习目标物体的特征，并使用这些特征来跟踪目标。深度学习模型将使用前一阶段获得的信息和当前帧图像来实现目标跟踪，其中最常用的方法是使用卷积神经网络。基于深度学习的图像跟踪技术具有很高的鲁棒性和精度，但需要大量的训练数据和计算资源。另一类发展出的技术是基于核相关滤波的图像

跟踪技术，这是一种基于模板匹配的图像跟踪技术，它使用核相关滤波算法来计算目标模板与当前帧图像的相似度，并根据相似度来确定目标在下一帧图像中的位置。核相关滤波是一种基于信号处理的滤波算法，它可以对信号进行频域滤波处理，并且具有不变性、局部性、鲁棒性等优点。在基于核相关滤波的图像跟踪中，先将目标模板和当前帧图像分别进行傅里叶变换，然后计算它们的复共轭积，再将结果通过傅里叶逆变换转换回空间域，得到响应图。响应图上的峰值代表了目标在当前帧图像中的位置。基于核相关滤波的图像跟踪技术的优点在于，它能够自适应地学习目标的外观变化，并且具有较高的鲁棒性，能够在目标尺度变化、旋转、遮挡等情况下保持较好的跟踪效果。同时，该技术计算速度较快，可以实现实时跟踪。

1.2.2 研究意义

视觉物体跟踪是计算机视觉界的基本任务之一，基本流程是通过在第一帧中给出的图像信息，进行连续的目标定位。目前的发展阶段，主流的研究方向分为判别式相关滤波器和基于深度学习的滤波算法[5]。相较于常见的基于普通 PC 机搭建的视觉跟踪器，基于无人机平台的系统在设计中需要考虑到了实际应用中的诸如能耗控制、算力限制、图传延迟等客观限制性因素，以及无人机操作人员的实际操控方式，包括整套系统的部署的便携性，操作逻辑的合理性和较高的可靠性等。本文的系统在设计过程中充分考虑到以上诸多限制，选用低功耗硬件设备配合算法优化，更好地适应用户使用时的电力供应，提高系统的续航能力；在保证检测精度和速度的前提下通过优化选择算法和硬件配置，尽可能地降低算力要求；通过考虑用户在操控无人机的同时使用本系统的场景，引入触控交互逻辑，更加契合实际需求；优化硬件模块使得本系统可以便于部署和携带，适用于不同的场景和任务需求。因此，本文设计的系统可以更好地适应基于无人机图传的平台，适应实际应用中的客观限制性因素和用户的实际操控方式，从而提高了系统的针对性和适应性。

1.3 本文结构及安排

本文由以下几个部分构成：

第1章　导论

本章主要介绍了本设计的内容及其研究背景，由无人机及其附属功能的发展引出基于无人机图传的图像目标低延时检测系统，在确定了研究意义之后对本文的内容进行了

介绍。

第2章　主体构成

本章主要介绍了本设计的主体硬件构成，包括空中端至地面段的图像传输模块、将图像传入检测系统部署至的移动平台的信号转换模块、部署系统的嵌入式计算平台以及供无人机操作人员进行操作的交互操控模块。

第3章　基于判别式相关滤波器的目标跟踪算法研究

本章介绍了基于判别式相关滤波器的目标跟踪算法，特别是在目标跟踪中的应用，包括其基本原理、优势和不足之处。

第4章　本系统的跟踪算法设计

本章介绍了常见的目标跟踪算法，并指出了其他主流算法的不可取之处，并提出了本系统的跟踪算法设计。

第5章　系统的部署调试

本章主要介绍了本系统的系统调试与部署，主要分为硬件部署和软件部署两个大类。至此，本系统的设计基本上完成。

第6章　总结与展望

本章总结了全文的研究内容和设计结果，并指出了所设计系统的一些不足之处，对后续研究进行了展望。

第 2 章　主体构成

2.1 本设计的主要硬件构成

为了实现将空中端的无人机获取的图像内容传输至地面端，并传入处理终端进行目标检测跟踪，最终将结果呈现给无人机操作人员这一整个流程，本系统的主要硬件构成包括以下 4 个模块：

（1） 图像传输模块：该模块的功能是将图像从空中端传输至地面端。

（2） 信号转换模块：该模块的功能是将传输至地面端的图像经过处理后传入检测系统部署至的移动平台。

（3） 嵌入式计算模块：该模块的功能是提供可部署系统的嵌入式计算平台。

（4） 交互操控模块：该模块的功能是供无人机操作人员进行交互操作。

2.2 图像传输模块

图像传输，是指通过一定的通信手段将视频信号传输到远程设备的过程。目前，图传主要有模拟图传和数字图传两个类别，由发射端、接收端和显示端三个部分组成[6]。在本系统的设计过程中，接收端的图像需要进行计算机视觉的处理，而不是直接呈现给无人机操作人员，故诸如 FPV 眼镜之类的显示端设备不属于图像传输模块中的硬件内容。

2.2.1 数字图传

数字图传是指利用数字信号传输技术将图像或视频信号进行数字化处理并传输的一种图传方式。目前市场上主流的数字图传方案包括 DJI 数字图传、Connex 数字图传、Caddx 数字图传等。

数字图传最大的优势在于可以更好地保持信号的清晰度。目前，数字图传的分辨率已经可以达到 4K 级别，极大地提高了无人机操作人员的沉浸感。然而，由于数字图像的传输原理涉及到转换、编码、压缩、解压缩等步骤以满足带宽限制，同一等级的数字图传的传输延迟大于模拟图传。本系统要求整个图像传输延迟尽可能地低，对于画面分辨率及质量的要求较为宽松，且考虑到消费级产品的预算，故本系统考虑使用模拟图传而不是数字图传。

2.2.2 模拟图传

与数字图传不同，模拟图传将无人机摄像头获取的实时视频信号转换为无线电信号，主要使用 5.8 GHz 射频频段进行传输，并将无线电信号再次转换为视频信号进行显示。目前市场上主流的模拟图传方案包括 Fatshark 模拟图传、Runcam 模拟图传、Blacksheep 模拟图传。

模拟图传的基本流程包括摄像头采集图像、调制、功放、传输、接收、解调和显示等。对于本系统而言，模拟图传最大的优势在于图像传输延迟较低。有别于使用 FPV 眼镜或者接收显示一体机等设备直接监控视频流，本系统在接收图像后需要引入计算机视觉进行目标检测跟踪。取决于终端设备的算力和算法的选择，这部分操作产生的延迟是相当可观且不可避免的。因此，图像传输模块的选择应满足尽可能低的延迟。模拟图传非常适用于 FPV 飞行、竞速和其他需要高实时性的应用场景，是本系统图像传输模块的不二之选。

2.2.3 图传套装的选择

由 2.2.2 知，本系统采用使用模拟图像传输方式的发射机和接收机。选择适合本系统的收发机需要考虑频段、信道、视频格式、接收距离、信号质量等因素。

表 2.1　RC832H 数据表

Table 2.1 RC832H Datasheet

规格参数	描述
工作频率	5.8 GHz
接收频点	48 点
电压输入	8～26 V
工作电流	200 mA
天线阻抗	50 Ω
视频制式	NTSC / PAL 自动

表 2.1 所示为一款型号为 RC832H（图 2.1）的 2.4 GHz 无线视频接收机的规格参数。该无线视频接收机是一款高性能的无线视频接收设备，广泛应用于航模、FPV 飞行、无人机拍摄等领域。它的稳定性和易用性深受用户好评，是许多航模爱好者和专业飞行员的首选之一。本系统采用 RC832H 作为图像传输模块的接收器，搭配 TS832 发射模块使

用，这一套图传在规格参数和实际性能上均满足要求。

图 2.1　RC832H 无线视频接收机

Fig. 2.1　RC832H wireless video receiver

在后期系统实际部署至用户使用时，同等甚至更高规格的无线视频接收机型号可作为替代选择。

2.3 信号转换模块

由 2.2 的内容可知，本系统确定了将图像从空中端传输至地面端的图像传输模块。下一步流程需要将图传接收机接收的图像传入至计算机视觉处理设备。

图 2.2　东芝 TC358840XBG

Fig. 2.2　Toshiba TC358840XBG

深入调查目前主流的消费级解决方案，本设计采用了一颗型号为 TC358840XBG 的东芝 CMOS 数字集成电路芯片（图 2.2）。该芯片属于桥接芯片的一种，用于将移动设备的移动显示接口转换成高清晰度多媒体接口输出，广泛应用于智能手平板电脑、笔记本电脑等移动设备中。

表 2.2　TC358840XBG 数据表

Table 2.2 TC358840XBG Datasheet

规格参数	描述
厂商	Toshiba
芯片类型	桥接芯片
支持分辨率	最高可达 4K / 30fps
最大时钟频率	297 MHz
内置 RAM	128 MB DDR2 SDRAM
供电输入	Core: 1.15V MIPI D-PHY: 1.2V I/O: 1.8V, 3.3V HDMI: 3.3V APLL: 3.3V
工作温度范围	-20℃至 85℃
封装尺寸	FBGA 7mm × 7mm
引脚间距	0.65mm
典型功耗	1920×1080 @60 fps: 420 mW 2560×1600 @60 fps: 504 mW 3840×2160 @30 fps: 520 mW

如表 2.2 所示，为该芯片的详细参数。该芯片支持 HDMI 至 CSI-2 的转换，由 2.4 的内容可知，本系统选择的嵌入式计算模块树莓派 4B 型号板载 CSI 接口。CSI 接口，即 CMOS Serial Interface，可以实现原生的相机设备支持。相较于传统的 USB 采集卡方式，使用该芯片理论可以承载高达 7.2 Gbps 的视频流，在保证分辨率和帧率的情况下延迟大大降低。

此外，由表 2.2 中的参数可知，该芯片具有功耗低和尺寸小的特点，有助于提升本系统的续航能力和部署便携性，符合本设计的实际需求。

2.4 嵌入式计算模块

该模块是计算机视觉的计算模块，负责进行图像处理和输出，以及与无人机操作人

员的交互操作控制。选择嵌入式计算平台需要根据项目需求、功能要求、性能和预算等因素进行综合考虑。目前主流的嵌入式计算平台包括 Arduino、NVIDIA Jetson、Raspberry Pi 等。

2.4.1 Arduino

Arduino 是一款基于开源硬件和软件的嵌入式计算平台，由意大利开发。它具有小巧、低功耗、易于使用和低成本等特点，适用于制作电子产品和控制系统。然而，Arduino 并不是最佳的图像处理平台，它的处理能力和存储容量都有着较大的限制，难以支撑连续的图像数据输入和复杂的图像处理，对于本系统需求的图像处理任务难以胜任。

2.4.2 NVIDIA Jetson

Nvidia Jetson 是一系列基于 ARM 架构的嵌入式计算平台，主要用于人工智能和机器学习等应用。Jetson 平台由 Nvidia 公司推出，旨在提供高性能、低功耗、易于开发的嵌入式计算解决方案。Jetson 平台基于 Nvidia 的 GPU 技术，具有出色的图形处理和并行计算能力，可以处理大量的数据和复杂的算法，支持深度学习、计算机视觉、自动驾驶、机器人等应用。

参考相关基于深度学习算法所搭建的平台，Jetson Nano 似乎是一个较为合适的选项。在与本系统的需求进行对比分析后发现，该嵌入式平台不能较好地契合需求。首先，本系统是与无人机紧密结合的系统，在考虑便携性的同时，地面端整机功耗也要严格控制，以保证在使用移动电源供电的使用场景下本系统拥有可观的续航能力。其次，本系统面向的用户在使用无人机时追踪的目标具有多样性和独特性。近年来，以 YOLO 系列算法为代表的深度学习目标检测算法被证实在 Jetson Nano 上运行有着可观的效率和精度。但是，该类算法需要制作特定的数据集，不能较好地满足无人机用户的跟踪检测需求，且检测效率普遍不及采用判别式相关滤波器的检测算法。

此外，Nvidia Jetson 价格高昂，考虑到实际部署的成本等原因，本系统在开发阶段选用其他嵌入式计算模块进行验证。在用户接受的范围内，后期可以使用更为高端的嵌入式计算模块提升系统的计算能力。

2.4.3 Raspberry Pi

树莓派是由树莓派基金会推出的一系列低成本、低功耗的单板计算机，其设计目标是为学习和 DIY 项目提供一个开放的平台。以树莓派 4B 4GB 版本为例，首先，它具有较强的计算性能和图形处理能力。它搭载的 Broadcom BCM2711 处理器包括四个 ARM Cortex-A72 核心，可以提供 1.5GHz 的处理器速度，此外还搭载了 4GB LPDDR4 内存，可以提供足够的计算资源和图形处理能力。其次，树莓派在确保整个系统的便携性的同时具备可扩展性。树莓派支持多种编程语言，如 Python、C、C++ 等，在系统的开发阶段可以根据需求和功能进行选择。此外，树莓派接口丰富，支持多种外设和扩展，如相机、键鼠、显示器、各类传感器等，其搭载的 CSI 接口具有较高的性能和可扩展性，支持多种不同的相机模块，并可以通过特殊配置来实现不同的信号源输入。最后，选择树莓派也可以提高研究效率和成本效益。相对于 Jetson Nano 等其他平台而言，树莓派具有较低的价格，更加经济实惠，同时树莓派的原生系统具备极佳的兼容性和易用性，拥有丰富的社区支持和开发资源，对用户使用而言可定制化程度更高，对系统后期开发潜力提高。

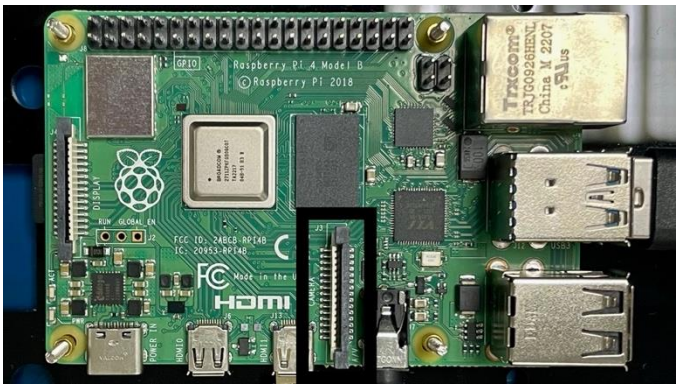

图 2.3　树莓派 4B 型号及其 CSI 接口（黑框处）

Fig. 2.3　Raspberry Pi 4B model and its CSI interface (at black box)

考虑到树莓派的运算性能以及所具备以上所述的优势，本系统选择树莓派 4B 作为部署平台。

2.5 交互操控模块

考虑到用户使用本系统时的实际情况，使用者在使用本系统的同时需要进行无人机的操控。将本系统与飞控系统的控制端较好地结合是本模块的首要目的。在与飞控团队进行深度交流后我们积极讨论，一致同意触控屏是最为理想的交互方式。本系统搭载一块 7 英寸标准清晰度触控屏幕，可与配套开发的固定支架结合使用，兼容不同型号的无

人机遥控器手柄。

在不断优化交互逻辑后，本系统实现了让用户单手触控操作触发所有功能，极大地降低了用户在使用本系统的同时对无人机进行操控的影响。

2.6 本章小结

本章主要介绍本设计硬件相关的组成模块，包括图像传输模块、信号转换模块、嵌入式计算模块、交互操控模块四个部分。包括每个模块的主要职责、目前主流的解决方案的优缺点横向对比、针对用户使用场景的定制化需求，最后在每个模块的结尾介绍了本系统所选择的方案。

第 3 章　本系统的跟踪算法设计

3.1 常见的目标跟踪算法介绍

几乎所有涉及到与目标跟踪算法相关的研究时，其关注的问题的本质是在给定一个视频中的初始帧中的一个目标实例后，如何在后续帧中准确地检测和跟踪该目标实例。就目前的发展现状而言，目标跟踪的应用场景非常广泛，不仅限于在本系统中作为核心功能发挥作用，在包括监视、自动驾驶、智能辅助和机器人控制等方面的应用大放异彩。该部分将介绍常见的目标跟踪算法，其中包括基于深度学习、基于神经网络和基于相关滤波的算法。

3.1.1 基于深度学习的目标跟踪算法

基于深度学习的目标跟踪算法是当前研究领域中最活跃的研究方向之一。深度学习在目标跟踪中的应用包括基于深度学习的目标检测和跟踪联合算法以及基于深度学习的目标跟踪方法。目前常见的基于深度学习的目标跟踪方法有 SiamFC、SiamRPN 和 SiamMask 等。这些方法将目标的跟踪问题转化为一个搜索问题，通过使用神经网络学习目标表征，以实现在视频序列中精确定位和跟踪目标。虽然这些算法在准确性和速度方面都取得了很好的表现，但深度学习算法的缺点包括对计算资源的要求高、对数据量的要求高以及对超参数的敏感性。

3.1.2 基于神经网络的目标跟踪算法

基于神经网络的目标跟踪算法有很多种，其中一种比较著名的算法是 YOLO（You Only Look Once）。YOLO 是一种基于深度卷积神经网络的目标检测算法，可以同时检测多个目标，并将其框定在图像中[10]。

YOLO 算法采用单一神经网络将整个图像作为输入，然后将该图像划分为不同的网格单元，每个网格单元预测多个目标的边界框和类别。通过使用卷积层和全连接层，YOLO 算法能够对输入图像进行端到端的处理，并输出目标的检测结果。在目标跟踪中，YOLO 算法可以通过对每个帧中的目标进行检测，并利用跟踪算法对目标进行跟踪，以实现在视频序列中精确定位和跟踪目标。

与其他基于神经网络的目标跟踪算法相比，YOLO 算法具有更快的检测速度和更高

的检测精度。需要注意的是，YOLO 算法通常需要使用 GPU 进行加速训练，因为深度学习模型的训练需要大量的计算资源和时间。此外，使用 YOLO 算法进行目标检测需要先训练一个适合特定任务和应用场景的模型。YOLO 算法通常需要大量的标注数据来进行训练，以学习目标的特征和上下文信息，从而在图像中进行准确的目标检测。

3.1.3 基于相关滤波的目标跟踪算法

基于相关滤波的目标跟踪算法是目标跟踪领域最具代表性的算法之一[11]。这些算法利用相关滤波来计算每个视频帧中目标的位置，并利用样本的特征和模板进行匹配。这些算法中的一些包括 KCF、CSR-DCF 和 Staple 等。这些算法具有较高的跟踪精度和计算速度，但它们对噪声和运动模糊等因素较为敏感。

3.2 基于神经网络的目标跟踪算法的不适用性分析

由前文可知，YOLO 算法是目前热度极高的基于神经网络的目标跟踪算法。基于 YOLO 算法的应用在各个领域都有所涉猎。遗憾的是，在分析本系统的使用场景和需要实现的功能之后，该算法并不是最为适合的选择。本章将从算法实现、实时性、算法可解释性等方面，究其原因详细论述。

3.2.1 算法实现

相比于传统的目标检测算法，YOLO 算法需要进行复杂的卷积计算，需要较高的计算资源，包括 GPU、显存、内存等。如果硬件资源限制，可能会导致算法无法运行或者运行缓慢。在确保系统具备可观的续航能力和可批量部署的经济性时，过高的硬件性能要求往往得不偿失。

表 3.1　不同版本的 YOLO 检测器在不同硬件设备上的 FPS[12][13][14][15]

Table 3.1 FPS of different versions of YOLO Detector on different hardware devices

版本	输入分辨率	GPU	FPS
YOLOv1	448×448	NVIDIA GTX 1080 Ti	45
YOLOv2	416×416	NVIDIA Titan X	40
YOLOv3	416×416	NVIDIA GTX 1080 Ti	30 ~ 60
YOLOv4	608×608	NVIDIA Tesla V100	65 ~ 70

YOLOv5	640×640	NVIDIA Tesla V100	140 ~ 150
YOLOv7	-	NVIDIA Tesla V100	150

由表 3.1 可知，随着版本的迭代，YOLO 算法的 FPS 逐渐满足了实时性方面的要求。虽然目前 YOLO 算法在保证一定精度的前提下达到了较为理想的速度，但是表中所示的 FPS 值是针对特定的硬件配置和输入图像大小达成的，实际的 FPS 值会因为硬件设备、输入图像的大小和分辨率等因素而有所不同。就本系统的使用场景而言，使用功耗极高且体积庞大的高性能独立 GPU 是极为不现实的。本文在 2.4.2 中提及的 Jetson Nano 内置 NVIDIA Maxwell GPU，根据官方介绍，其性能大概相当于一个传统桌面 PC 上的 GeForce GT 1030 显卡。显然地，与表 3.1 中所使用的 GPU 依然存在较大的性能差异。

3.2.2 实时性

在实时性方面，虽然 YOLO 算法通过不断迭代达到了较为理想的速度，但是使用 YOLO 算法进行目标检测通常需要先训练一个适合特定任务和应用场景的数据集或预训练权重。YOLO 算法通常需要大量的时间进行训练，以学习目标的特征和上下文信息，从而在图像中进行准确的目标检测。在训练过程中，YOLO 算法通常需要使用 GPU 进行加速训练，因为深度学习模型的训练需要大量的计算资源和时间。在本系统中的用户使用场景中，图像目标具备多样性的特点，对于数据集以外的检测目标，YOLO 算法未曾进行标注数据和训练，几乎无法满足实时性的需求。

3.2.3 算法可解释性

在算法可解释性方面，YOLO 算法也存在一些劣势。由于 YOLO 算法是基于神经网络的黑盒模型，其模型结构较为复杂，很难直观地解释算法的检测过程和判断依据。此外，YOLO 算法对于一些特殊场景的检测效果也比较难以解释，例如在遮挡目标和复杂背景下的检测效果。

3.3 相关滤波器的基本概念

3.3.1 图像滤波器

图像滤波器是一种对图像进行处理的技术，其主要目的是去除图像中的噪声和增强图像的特征。常用的图像滤波器包括均值滤波器、高斯滤波器、中值滤波器等。这些滤

波器可以应用于灰度图像、彩色图像以及视频序列等。

3.3.2 线性滤波器和非线性滤波器

根据滤波器的特性，图像滤波器可以分为线性滤波器和非线性滤波器两个类型。

线性滤波器是数字图像处理中常用的一种滤波器，它可以将输入图像上的局部区域内的像素进行加权求和，从而得到输出像素值。由于线性滤波器的计算过程可以用矩阵乘法来表示，因此计算速度快且易于实现。然而，线性滤波器的处理效果对于图像中的高频细节信息如纹理和边缘等并不理想。

相对地，非线性滤波器在数字图像处理中也很常用，它们的滤波操作并不是基于线性变换的。常见的非线性滤波器有中值滤波器和最大值滤波器，它们通过对输入图像局部区域内的像素进行排序，从而得到输出像素值。非线性滤波器不依赖于像素值的大小关系，因此可以保持图像边缘并消除噪声等，但相应地需要进行排序操作，计算复杂度相对较高。

3.3.3 相关滤波器的基本概念和原理

相关滤波器是一种基于模板匹配的非线性滤波器，其基本原理是通过比较图像上某个窗口内的像素值和一个预先定义好的模板来实现滤波操作。具体来说，相关滤波器的输出像素值是由输入图像上的一个局部区域和一个模板卷积得到的。

在相关滤波器中，模板是滤波器的核心部分，通常是一个大小固定的矩形或圆形区域，用于描述所需特征。模板中的像素值可以根据具体应用场景来设置，例如可以设置为高斯分布的权值，以便更好地匹配图像中的特征。

滤波器的输出值可以表示为两个向量之间的相关系数，也就是输入图像中的像素向量和模板向量之间的相似度。根据相关系数的大小，可以确定输入图像中哪些像素是与模板最匹配的。相关滤波器通过将模板在输入图像上移动来处理整个图像，输出的像素值替换了原图像中对应位置的像素值，从而实现滤波操作。

相关滤波器具有许多优点。例如可以通过模板来调节滤波器的响应特性，适应不同的应用场景。针对于实时目标检测算法而言，相关滤波器的计算复杂度较低，可以快速地对大量图像进行处理。在延迟控制方面具有强大的优势。

需要注意的是，模板的大小和形状对滤波器的性能影响较大。较小的模板通常适用于处理具有明显边缘特征的图像，而较大的模板则适用于处理相对平滑的图像。

3.4 判别式相关滤波器的基本原理

3.4.1 判别式相关滤波器的概念

判别式相关滤波器（Discriminative Correlation Filter，DCF）是一种基于统计学习理论的滤波器，用于目标跟踪、目标检测等计算机视觉领域的任务[7]。相比于传统的相关滤波器，判别式相关滤波器能够更好地适应不同的图像场景和目标特征。它是通过学习训练集中的样本特征来提高滤波器的准确性和泛化能力。

3.4.2 判别式相关滤波器的实现

判别式相关滤波器的实现基于贝叶斯决策理论，其输出像素值是根据输入图像的特征向量和模板向量之间的相关系数来计算的[8]。首先，假设待跟踪目标在当前帧的位置为 x，我们希望通过在此位置上的滤波器响应来判断目标是否出现在该位置。对于一个大小为 $w \times h$ 的滤波器 F，其响应 r 可以通过将滤波器与当前帧图像 I 在位置 x 处进行卷积得到：

$$r(x) = \sum_{i=1}^{w} \sum_{j=1}^{h} F(i,j) \cdot I(x+i, y+j) \tag{3-1}$$

其中，$I(x+i, y+j)$ 表示当前帧图像中以 $(x+i, y+j)$ 为左上角顶点的大小为 $w \times h$ 的图像块。

接下来，我们需要定义一个代价函数 J 来度量当前帧中目标出现在不同位置 r 的概率。在判别式相关滤波器算法中，常用的代价函数为平方误差代价函数：

$$J(x) = \sum_{i=1}^{w} \sum_{j=1}^{h} | F(i,j) \cdot I(x+i, y+j) - G(i,j) |^2 \tag{3-2}$$

其中，G 是目标模板，是在跟踪开始时在目标位置处获取的一张图像块，用于表示目标的外观特征。通过最小化代价函数 $J(x)$，可以得到最有可能的目标位置 x，即：

$$x = argmin_x J(x) \tag{3-3}$$

将代价函数 $J(x)$ 化简后，可以得到如下形式：

$$\begin{aligned} J(x) = & \sum_{i=1}^{w} \sum_{j=1}^{h} | F(i,j) |^2 \cdot | \hat{I}(x+i, y+j) |^2 \\ & -2Re\{\sum_{i=1}^{w} \sum_{j=1}^{h} F(i,j)^* \cdot \hat{G}(i,j) \cdot \hat{I}(x+i, y+j)\} \end{aligned} \tag{3-4}$$

其中，$\hat{I}$ 和 $\hat{G}$ 分别表示 I 和 G 的离散傅里叶变换：

$$\hat{I}(u,v) = \sum_{i=1}^{w}\sum_{j=1}^{h} I(i,j)\cdot e^{-j2\pi(\frac{ui}{w}+\frac{vi}{h})} \tag{3-5}$$

$$\hat{G}(u,v) = \sum_{i=1}^{w}\sum_{j=1}^{h} G(i,j)\cdot e^{-j2\pi(\frac{ui}{w}+\frac{vi}{h})} \tag{3-6}$$

在判别式相关滤波器算法中，我们认为 I 和 G 是平稳随机过程，因此它们的离散傅里叶变换是平稳的。根据这个假设，我们可以将滤波器 F 表示为一个复值矩阵，其实部和虚部分别对应了滤波器的实部和虚部：

$$F(u,v) = \rho(u,v)\cdot e^{i\theta(u,v)} \tag{3-7}$$

其中，$\rho(u,v)$ 和 $\theta(u,v)$ 分别表示 $F(u,v)$ 的幅度和相位。为了得到最小的代价函数 $J(x)$，我们需要求解 $\rho(u,v)$和 $\theta(u,v)$。通过最小化 $J(x)$，我们可以得到 $\rho(u,v)$ 和 $\theta(u,v)$ 的更新公式：

$$\rho(u,v) = \frac{\sum_{i=1}^{w}\sum_{j=1}^{h}\hat{G}(i,j)\cdot\hat{I}^{*}(x+i,y+j)\cdot e^{-i\theta(u,v)}}{\sum_{i=1}^{w}\sum_{j=1}^{h}|\hat{I}(x+i,y+j)|^{2}} \tag{3-8}$$

$$\theta(u,v) = arg\{\sum_{i=1}^{w}\sum_{j=1}^{h}\hat{G}(i,j)\cdot\hat{I}^{*}(x+i,y+j)\cdot e^{-i\theta(u,v)}\} \tag{3-9}$$

其中，arg 表示求取复数的辐角。

最后，我们可以将滤波器响应 $r(x)$ 和代价函数 $J(x)$ 用一个 sigmoid 函数进行映射，得到最终的概率估计值 $p(x)$：

$$p(x) = \frac{1}{1+exp(-\alpha\cdot r(x)-\beta\cdot J(x))} \tag{3-10}$$

其中，α 和 β 是可以自定义的参数。根据概率估计值 $p(x)$，我们可以更新目标的位置，并继续跟踪目标。

综上所述，判别式相关滤波器的实现原理是通过最小化平方误差代价函数来估计目标的位置，利用离散傅里叶变换来加速计算，使用 sigmoid 函数将滤波器响应和代价函数进行组合，得到概率估计值，进而实现目标跟踪[9]。

3.4.3 判别式相关滤波器的训练过程

判别式相关滤波器的训练过程包括两个主要步骤：正样本的学习和负样本的抑制。

正样本是指与目标特征高度相关的图像区域，例如目标物体的边缘或纹理等。在学习阶段，判别式相关滤波器使用正样本来更新模板向量，使其能够更好地匹配输入图像中的目标特征。

负样本是指与目标特征无关的图像区域，例如背景或其他物体等。在训练过程中，判别式相关滤波器使用负样本来抑制与目标特征无关的图像区域，从而提高滤波器的鲁棒性和泛化能力。

判别式相关滤波器具有很好的性能和适用性，能够适应不同的图像场景和目标特征。在实际应用中，判别式相关滤波器通常采用在线学习的方式进行训练，即在跟踪过程中不断更新模板向量，以适应目标的运动和变形。其训练过程需要足够的正负样本，并且需要选择适当的特征描述符和模板大小来提高滤波器的准确性和泛化能力。

3.5 基于判别式相关滤波器的目标跟踪算法的适用性分析

3.5.1 目标跟踪的基本原理

目标跟踪是指在一个连续的视频序列中追踪一个特定的目标，这个目标可能是一个人、一个车辆，抑或是用户选定的某个特定物体。目标跟踪的基本目的是通过一系列的图像处理和模型匹配技术，实现对该目标在每一帧图像中的位置和形状的准确估计。其基本原理是在当前帧图像中搜索目标物体的位置，然后在每一帧图像中寻找与模型最相似的区域，从而实现目标的追踪。因此，快速和准确是衡量一个目标跟踪算法优劣的关键。

3.5.2 判别式相关滤波器在目标跟踪中的应用

判别式相关滤波器是一种常用的目标跟踪算法，其利用线性分类器来区分目标和背景，并自适应地更新模型以适应目标的运动和变形，从而具有良好的实时性和精度。该算法通过训练样本集来得到滤波器的模型参数，然后使用这些参数对新的图像进行滤波，以获取目标的位置信息。相比其他目标跟踪算法，判别式相关滤波器的主要优势在于其能够自适应地适应目标的运动和变形，并且实时性好。

3.5.3 判别式相关滤波器在目标跟踪中的优势和不足

综合来看，判别式相关滤波器在目标跟踪中存在以下的优势和不足：

优势：

- 精度高： 判别式相关滤波器可以对目标进行精确的跟踪，即使目标发生了形变、遮挡等变化，仍能够保持高精度。
- 实时性好：判别式相关滤波器具有较快的运行速度，能够在实时场景中进行目标跟踪。
- 可扩展性强：判别式相关滤波器可以通过不同的特征表示方法来适应不同的目标跟踪任务，具有较强的可扩展性。

不足：

- 尺度变化问题：判别式相关滤波器对目标的尺度变化比较敏感，对于尺度变化比较大的目标，跟踪效果可能不佳。
- 背景干扰：当背景与目标相似或者变化比较剧烈时，判别式相关滤波器的跟踪效果也可能不佳。
- 训练集样本不足：判别式相关滤波器的性能受训练集的影响较大，如果训练集中的样本不足或者不够典型，可能会影响跟踪效果。

综上，判别式相关滤波器在目标跟踪中有着较好的应用，但需要结合实际使用场景进行改进和优化，尽可能地减小算法的不足之处带来的负面影响，以提高跟踪效果和可靠性。

3.6 本系统使用的 CSRT 算法介绍

3.6.1 算法原理

CSRT 算法是一种基于相关滤波器和特征点选择的目标跟踪算法[16]。它将目标的图像特征分为两个部分：通道可靠性特征和空间可靠性特征。其中，空间可靠性特征是指利用了不同尺度的特征点对目标区域进行描述，以适应目标缩放和旋转变化。通道特征是指通过多通道特征提取，包括颜色、纹理等，以适应目标的外观变化和遮挡等情况。为了提高跟踪的准确性和鲁棒性，CSRT 算法还引入了自适应性和尺度不变性等机制。

基于 CSRT 算法的目标检测方法主要包括以下三个步骤：

- 初始化阶段：在初始化阶段，CSRT 算法首先需要选择一个初始的目标区域，并从该区域中提取目标的外观模型。具体而言，CSRT 算法将目标的外观模型

表示为一组滤波器，每个滤波器对应目标的一个特征通道。常用的特征通道包括颜色、纹理等。

- 跟踪阶段：在跟踪阶段，CSRT 算法使用学习到的外观模型对当前帧的图像进行滤波，得到一组响应图。然后，通过对响应图进行加权求和，CSRT 算法得到一个综合的响应图，用于确定目标在当前帧中的位置。
- 更新阶段：CSRT 算法将当前帧的图像分割成多个区域，并对每个区域内的像素进行分类，以确定该区域是否包含目标。然后，CSRT 算法使用包含目标的区域来更新目标的外观模型，以适应目标的变化。

3.6.2 算法实现

CSRT 算法中，引入了通道可靠性和空间可靠性的概念，用于提高跟踪的稳定性和准确性。

假设目标模板为 f，输入帧为 x，则目标在帧中的位置 p 可以表示为：

$$p = argmax_q H(q) \tag{3-11}$$

其中，$H(q)$ 为响应图像，表示在图像上的每个位置 q 上的相似度得分，即：

$$H(q) = \sum_i w_i \cdot R_i(q) \tag{3-12}$$

其中，w_i 是第 i 个通道的权重，$R_i(q)$ 是第 i 个通道的响应值，它们分别可以表示为：

$$w_i = \frac{1}{2}(1 + \alpha_i - \beta_i) \tag{3-13}$$

其中 α_i 和 β_i 是通道 i 的可靠性参数。当通道 i 的可靠性较高时，权重 w_i 更接近 1；当通道 i 的可靠性较低时，权重 w_i 更接近 0.5；

$$R_i(q) = \frac{\sum_u \sum_v k_i(u,v) \cdot g(x_{q+u,v}, f_{u,v})}{\sqrt{\sum_u \sum_v k_i(u,v)^2 \cdot \sum_u \sum_v g(x_{q+u,v}, f_{u,v})^2}} \tag{3-14}$$

其中，$k_i(u,v)$ 是第 i 个通道的相关滤波器核，$g(x_{q+u,v}, f_{u,v})$ 表示输入帧 x 中位置 $(q+u,v)$ 和目标模板 f 之间的相似度。该式表示在第 i 个通道上，计算输入帧 x 中位置 $(q+u,v)$ 与目标模板 f 的相似度，然后乘以该通道的相关滤波器核 $k_i(u,v)$，最后将所有位置的乘积加起来得到响应值 $R_i(q)$。在分母中，除以的是所有位置的相似度值平方和和相关滤波器核的平方和的开方，目的是进行归一化，使得响应值的大小不会受到输入帧的亮度等因素的影响。式中 $g(x_{q+u,v}, f_{u,v})$ 可以表示为：

$$g(x_{q+u,v}, f_{u,v}) = \frac{\sum_{i=1}^{d} (x_{q+u,v,i} - \mu_i)(f_{u,v,i} - \nu_i)}{\sqrt{\sum_{i=1}^{d} (x_{q+u,v,i} - \mu_i)^2 \cdot \sum_{i=1}^{d} (f_{u,v,i} - \nu_i)^2}} \tag{3-15}$$

其中，d 是输入图像的通道数，μ_i 和 ν_i 分别是输入图像和目标模板在第 i 个通道上的平均值。

3.6.3 CSRT 算法参数设置

CSRT 算法中包含可以调节的参数，它们会对算法的跟踪性能和运行速度产生影响。其中最重要的几个参数包括高斯核函数的标准差 sigma、L2 正则化项的权重 l2_regularizer、HOG 描述子中每个像素的梯度方向数目 hog_orientations、扩展目标边界框大小的像素数量 padding，以及模板大小 template_size 等。

进行算法的参数设置这一步骤需要在建立本系统的跟踪算法后、跟踪开始之前进行。首先，按照 M. Danelljan 等人的参数进行设置，作为初始参数，在 PC 机上导入一段测试视频序列并运行跟踪器，记录跟随性能和运行速度。然后引入网格搜索优化算法，对最佳参数设置进行搜索。最后，对于用于测试的视频序列使用已经优化的参数设置。

3.6.4 CSRT 算法参数调优策略

在进行参数调优时，需要结合本系统的实际需求制定调整策略。由于低延时是本系统追求的主要目标之一，本系统的算法调试将跟踪准确率和相应速度进行加权平均，作为该算法改进的综合性能指标。在牺牲一定的跟踪准确率的条件下得到更高的 FPS，即算法的响应速度，以满足本系统的设计要求。

3.7 本章小结

本章主要介绍了本系统的跟踪算法设计。首先是常见的目标跟踪算法的介绍以及基于以 YOLO 算法为代表的神经网络算法的不适用性。然后介绍了相关滤波器的基本概念和原理，包括图像滤波器、线性滤波器和非线性滤波器等。之后，本章详细介绍了判别式相关滤波器的基本原理，包括其概念、数学表达式和训练过程，以及判别式相关滤波器在目标跟踪中的应用，包括其基本原理、优势和不足之处。在此基础上介绍了 CSRT 算法，包括它的原理和实现。以及本系统在使用 CSRT 算法时，在算法的基础上对参数进行优化，将响应速度的权重调高并对最佳参数设置进行搜索的参数调优策略。最后将

该算法与本系统的实际应用相结合进行优化调整，以获得更快的速度和更好的跟踪效果。

第 4 章　系统的调试与部署

4.1 系统调试

4.1.1 CSRT 算法可调参数

在 CSRT 算法中，可供调节的参数如表 4.1 所示。

表 4.1　CSRT 算法可调参数

Table 4.1 Adjustable parameters of CSRT algorithm

数据类型	参数名称	描述	默认值
int	admm_iterations	ADMM 算法的迭代次数	60
int	background_ratio	前景掩码计算的背景比例	0.7
float	cheb_attenuation	Chebyshev 窗函数的衰减因子	0.0
float	filter_lr	过滤器更新的学习率	0.02
float	gsl_sigma	高斯空间窗口的 Sigma 值	0.5
int	histogram_bins	颜色直方图的箱数	16
float	histogram_lr	直方图更新的学习率	0.04
float	hog_clip	HOG 特征的剪裁值	0.2
float	hog_orientations	HOG 描述符的方向数	9
float	kaiser_alpha	Kaiser 窗函数的 Alpha 值	1e-4
int	num_hog_channels_used	使用的 HOG 通道数	10
int	number_of_scales	尺度估计的尺度数	33
float	padding	搜索窗口中目标周围的填充量	3.0
float	psr_threshold	峰值信号与旁瓣比（PSR）的阈值	3.0
float	scale_lr	尺度估计更新的学习率	0.025
float	scale_model_max_area	尺度估计模型的最大面积	512.0
float	scale_sigma_factor	尺度估计的 Sigma 因子	0.25
float	scale_step	尺度估计的尺度步长	1.02
float	template_size	用于跟踪的模板大小	128.0
bool	use_channel_weights	是否使用通道权重进行特征计算	true

bool	use_color_names	是否使用颜色名称进行特征计算	true
bool	use_gray	是否使用灰度图像	false
bool	use_hog	是否使用 HOG 特征	true
bool	use_rgb	是否使用 RGB 图像	true
bool	use_segmentation	是否使用分割进行跟踪	false
float	weights_lr	特征权重更新的学习率	0.95
std::string	window_function	使用的窗口函数类型："hann"、"cheb"或"kaiser"	hann

表 4.1 中的参数影响 CSRT 算法在目标跟踪过程中的特征提取、模型更新和尺度变换等方面，以提高目标跟踪的准确性和鲁棒性。根据具体的应用场景和目标特点，可以调整这些参数来优化算法的性能。

4.1.2 运用调试工具

在表 4.1 中所列参数中，包含对性能影响较为显著的几个参数。例如，window_function 控制使用的窗口函数类型，不同的窗口函数会影响搜索窗口的形状，从而影响跟踪器的性能；histogram_bins 控制颜色直方图的箱数，较大的值可以提高颜色直方图的精度，但可能会增加计算成本；admm_iterations 控制 ADMM 算法的迭代次数，较大的值可以提高跟踪器的鲁棒性，但会增加计算成本。

在使用 CSRT 算法进行目标跟踪的 Python 中，引入网格搜索优化算法，定义参数搜索空间和评估函数，使用 GridSearchCV 类进行网格搜索。在评估函数中，通过计算出跟踪速度，并以此为基准对函数进行评估。

4.1.3 调试常见问题

网格搜索会在参数搜索空间中尝试不同的参数组合，并根据评估函数计算出每组参数的评分，最终输出最优参数和对应的评分。在本系统的算法调试过程中，评估函数使用速度作为评分，因此最优参数应该是使得跟踪速度最快的参数组合。然而，使用不同的数据集和参数组合会得到不同的评分结果，在保证速度的前提下，所设置的参数也应满足可靠的准确率。此外，自定义评估方法的计算复杂度也会影响参数搜索优化的效率，因此还需要权衡计算复杂度和评估精度的关系。

4.2 系统部署

4.2.1 硬件连接

由 2.1 内容可知，本系统的硬件部分包括图像传输模块、信号转换模块、嵌入式计算模块、交互操控模块四个部分。模块间的工作流程图如图 4.1 所示。

首先由图像传输模块接收由无人机端的图像发射设备发出的视频信号，并将该信号传入信号转换模块。随后，信号转换模块将信号格式进行转换，传入嵌入式计算模块。之后，嵌入式计算模块对图像进行处理，输出至交互操控模块呈现给用户，同时时刻准备响应来自交互操控模块的输入动作。

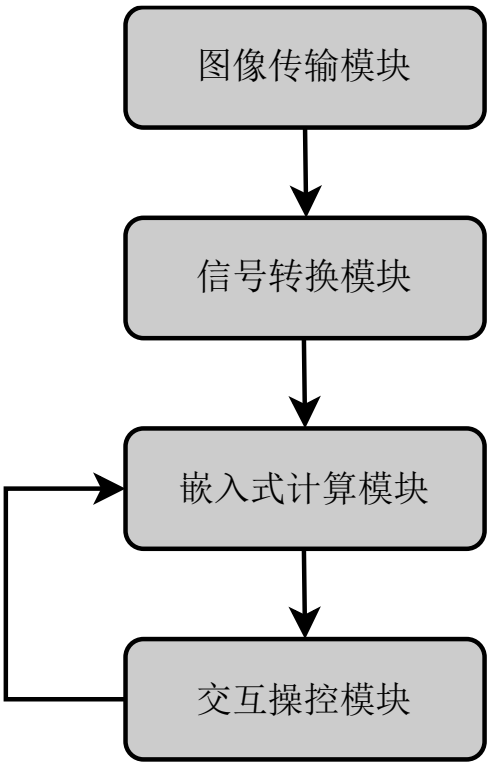

图 4.1　硬件工作流程图

Fig. 4.1　Hardware workflow chart

4.2.2 硬件环境搭建

由于本系统所使用的算法需要在树莓派上运行，在进行代码移植前，需要先搭建树莓派端硬件环境。首先是下载安装树莓派官方镜像。为了最大程度发挥树莓派的性能，此处选择的是 64 位版本的官方镜像。由于中国大陆网络限制，在树莓派上部署 OpenCV 存在一定的复杂性。我们选择使用 apt-get 命令安装 OpenCV 所需要的依赖库，例如

libjpeg-dev、libpng-dev、libtiff-dev 等，然后在 OpenCV 官网上下载最新版本的源代码并解压。之后在本地使用 cmake 进行编译配置，然后使用 make 命令进行编译，最后使用 make install 命令进行安装 OpenCV 源代码。安装完成后使用 python 进行 OpenCV 的测试，确认没有报错之后，硬件环境配置成功。

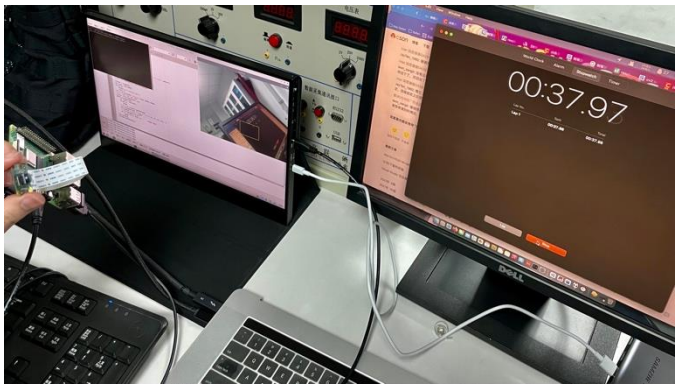

图 4.2　硬件环境测试

Fig. 4.2　Hardware environment test

如图 4.2 所示，为硬件环境搭建完成后的测试过程，此时树莓派已满足本系统的运行条件。

4.2.3 代码生成与移植

如图 4.3 所示，为本系统的主程序流程图。完成本系统所使用的跟踪算法的代码之后，下一步需要开发的就是本系统的主程序代码。主程序的运行逻辑是，当主程序开始运行后，调用 cv2.VideoCapture()方法，将树莓派从信号转换模块接收到的视频流传入程序，并使用一个 while 循环不断调用 cv2.imshow()方法保持图像显示。与此同时，程序准备响应用户的输入，当用户框选出目标后，调用跟踪器进行初始化操作。之后根据主程序设置的计数器判断当前帧是否为关键帧，若是，则将当前图像传送至跟踪器进行更新，并根据跟踪器回报的目标坐标信息在主窗口绘制目标位置并再下一个关键帧重复更新操作。再编写函数调用本系统使用的跟踪算法。与此同时，程序准备响应用户的输入，当用户需要停止跟踪时，主程序清除当前传入跟踪器的目标坐标信息，使跟踪器处于待命状态不进行使能，同时主程序继续不间断地显示图像并准备响应用户框选目标的输入。

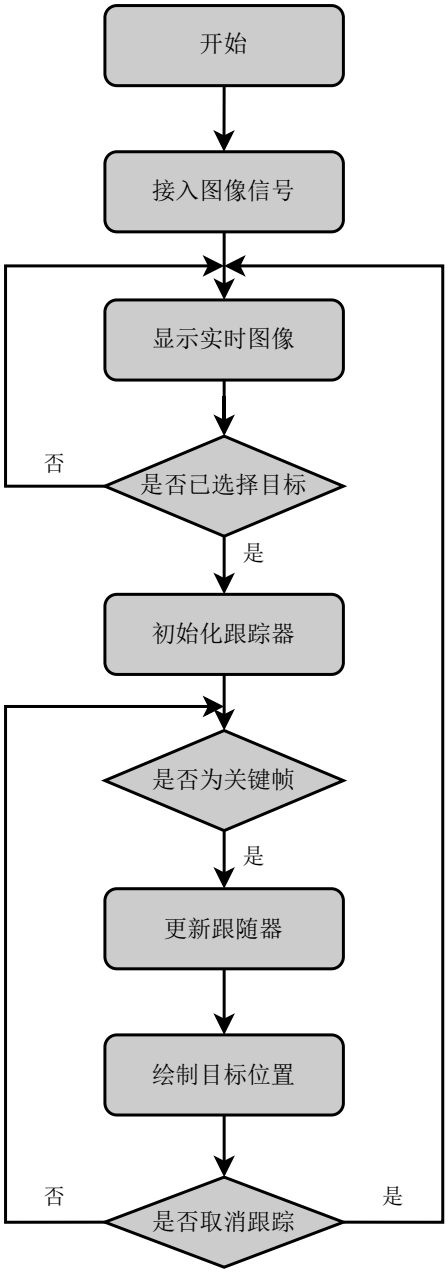

图 4.3　主程序流程图

Fig. 4.3　Main program flowchart

在 PC 机上进行主程序开发后，需要将其移植到树莓派并进行相应的适配。首先要传入视频流。在 Linux 操作系统中，通常/dev/video(0)是第一个视频设备文件，也通常连接到主板上的第一个摄像头。但由于本系统的信号转换模块使用了一颗桥接芯片，我们需要在程序中重新指定所使用的视频设备。针对触控屏的适配工作中，我们发现在调试阶段，程序检测到的用户输入为键鼠输入操作，这与我们实际部署后的要求不符。为了解决这个问题，我们进行了相应的适配工作，将所有事件响应替换为适合触控屏的单点、滑动等操作。此外，由于显示器分辨率发生了改变，窗口大小以及相应的坐标转换算法也进行了相应的更新。

4.3 系统验证

在完成系统的调试和部署之后，本系统在实验室环境下进行了系统验证。

4.3.1 系统的可靠性验证

本系统的可靠性主要体现在目标跟踪的鲁棒性。本系统在进行目标跟踪的测试时，模拟了实际使用场景中的不同情况，并对本系统的跟踪能力进行了记录。

如图 4.4 所示，在测试过程中模拟目标快速运动的情况。在测试过程中，检测目标进行了快速的移动，体现在程序中的影响是目标的坐标在短时间内快读改变，同时目标模版发生了一定程度的旋转变形。在此情况下，本系统的跟踪器极为准确地绘制出了检测目标的位置，没有出现跟踪丢失或跟踪滞后的情况。

图 4.4　测试目标快速运动

Fig. 4.4　Test target fast movement

如图 4.5 所示，在测试过程中模拟目标受到遮挡的情况。在测试过程中，检测目标被处于前景的障碍物遮挡，体现在程序中的影响是传入跟踪器的图像突然发生剧烈变化，但没有运动的趋势。在此情况下，本系统的跟踪器依然对目标位置置信度最高的区域做出了判断，并准确地绘制出了检测目标的位置，没有出现跟踪丢失或坐标漂移的情况。

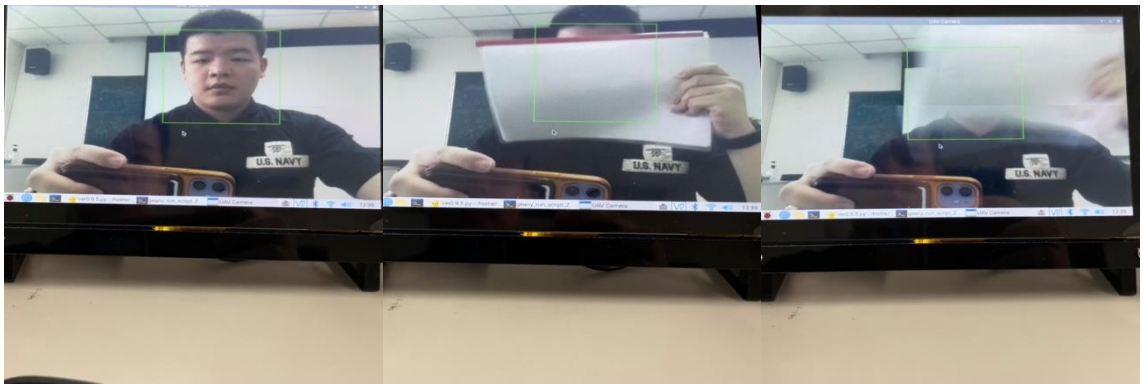

图 4.5　测试目标受到遮挡

Fig. 4.5　Test target obscured

如图 4.6 所示，在测试过程中模拟画面中同时存在多个相似目标干扰的情况。在测试过程中，检测目标周围出现了相同的物体进行干扰，两者处于静止、相对运动、并列以及重叠等状态。体现在程序中的影响是在跟踪器更新时会收到图像的干扰。在此情况下，本系统的跟踪器依然正确地判断出了原始检测目标，并准确地进行跟踪，没有出现跟踪错误的情况。

图 4.6　测试目标受到相似物体干扰

Fig. 4.6　Test target disturbed by similar objects

如图 4.7 所示，在测试过程中模拟画面背景光源特别强烈的情况。在测试过程中，检测目标所处的背景光源强烈或变化程度较大，体现在程序中的影响是影响跟踪器进行更新时的判断。在此情况下，本系统的跟踪器依然对检测目标的位置做出了正确的判断，并进行持续的追踪，没有出现跟踪丢失或跟踪滞后的情况。

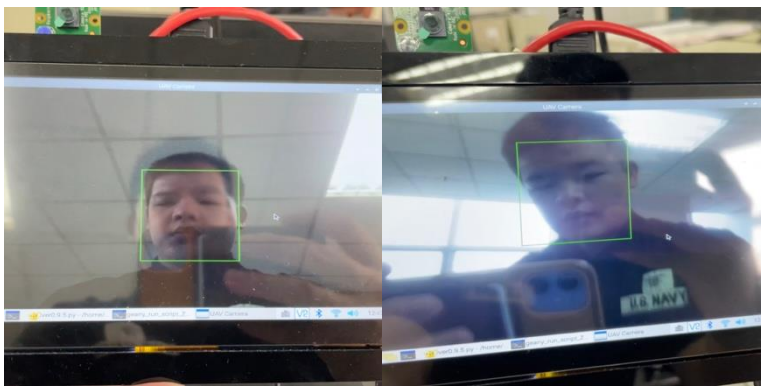

图 4.7　测试目标背景光照强烈变化

Fig. 4.7　Test target background light intensity change

经过以上详尽的测试模拟，我们对本系统在用户实际使用时，机载摄像头在空中环境可能遇到的各种情况进行了全面考虑。这些情况包括目标的快速运动、受到遮挡、相似物体的干扰以及背景光照的强烈变化等。通过实际验证的结果表明，本系统在应对这

些复杂场景时展现出了出色的跟踪鲁棒性。综上所述，我们可以充分验证本系统能够稳定、准确地进行目标跟踪，具有可靠性。

4.3.2 系统的实时性验证

本系统的延时产生主要产生于图像传输模块、信号转换模块和嵌入式计算模块。落实到具体环节上，体现在图像传输、图像格式转换、图像显示和检测跟踪算法四个环节。

如图 4.8 所示，为本系统在调试过程中接收图传图像，左侧为专用于 FPV 飞行使用的图传接收显示一体机“小飞手大师版”，右侧为本系统的触控显示屏。

图 4.8　小飞手大师版（左）与本系统（右）同时接收图像

Fig. 4.8　Hawk Eye (left) and our system (right) receive image simultaneously

如图 4.9 所示，为本系统在调试过程中调用摄像头显示图像测得的实时延迟。

图 4.9　本系统调用摄像头显示的实时延迟

Fig. 4.9　Real-time delay of our system calls camera display

如图 4.10 所示，为本系统在调试过程中调用摄像头显示图像同时进行目标检测时测得的实时延迟。

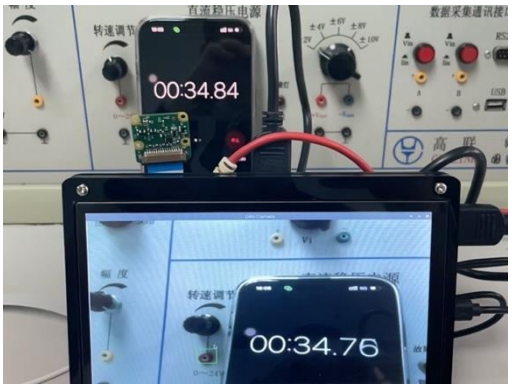

图 4.10　本系统调用摄像头进行目标检测的实时延迟

Fig. 4.10 Real-time delay of our system to call the camera for target detection

以上测试流程由各个小组成员重复进行操作，得到了数组延迟数据。经过计算取平均值作为本系统的延迟参考值。最终测试得到的结果是，图像传输模块的延迟在 30 毫秒左右；图像格式转换的延迟在 50 毫秒左右；图像显示的延迟在 40 毫秒左右；检测跟踪算法的延迟在 40 毫秒左右。

综上所述，本系统在验证阶段测得的总延迟在 160 毫秒左右，基本满足低延时的特性。

4.4 本章小结

本章主要介绍了本系统的系统的调试、部署及验证，首先是对 CSRT 算法的可调参数进行介绍，并重点研究了对于性能影响较大的部分参数。在此基础上，运用调试工具对进行改进，针对影响性能的参数进行优化调整。之后介绍了硬件部署的主要步骤，以及系统主程序的代码编写及主要的函数和调用。最后对本系统的性能指标进行验证，主要为可靠性和实时性两个方面，测试结果基本满足要求。至此，本系统的设计基本上完成。

第 5 章　总结

5.1 工作总结

本文设计了基于无人机图传的图像目标低延时检测系统，对其基本要求和使用场景的分析，以低延迟作为本系统的致力目标，从硬件设计到软件设计深度优化，最终通过安装与调试实现了本系统的运行。所做的具体工作可总结如下：

1) 第二章，介绍了本设计硬件相关的组成模块，包括图像传输模块、信号转换模块、嵌入式计算模块、交互操控模块四个部分。在详细介绍的部分，列举了目前主流的解决方案，在图像传输模块和嵌入式计算模块部分详细阐述了本系统在考虑用户使用场景、批量部署以及降低延迟方面的研究。

2) 第三章，介绍了本系统的跟踪算法设计。在前人研究的基础上，对于基于神经网络的 YOLO 算法进行了研究，根据这一算法的特性确定了其不可行之处。之后介绍了基于判别式相关滤波器的跟踪算法，基于基础的相关滤波器的基本概念和原理，研究了判别式相关滤波器作为一种重要的图像处理技术在计算机视觉领域的应用以及其适用性。在此基础上介绍了 CSRT 算法，在前人研究的基础上对其原理和实现进行了探讨，同时研究了该算法应用于本系统的可行性。所提出的算法在实时性和复杂度上具有极大的优势。基于前人的研究成果，对其优势和不足之处进行归纳。

3) 第四章，在前一章的基础上对基于 CSRT 算法的原理研究了该算法的可调参数，并分析得出其中对性能影响较为显著的几个参数。在 PC 机上部署该算法进行调试，以跟踪速度作为算法评分对该算法的参数进行优化，针对本系统的使用环境和特点进行改进。经过改进的算法更为契合本系统的要求。最后，进行本系统的硬件和软件的部署，并调试和验证本系统的实现。

5.2 问题与不足

在本系统的研发过程中，硬件模块使用的元器件较为固定，例如摄像头、图像传输模块、触控显示屏等。因此，软件参数的研究与优化针对的是本系统所使用的特定的分辨率、帧率及中央处理器处理能力。为确保本系统具备针对不同规格硬件的兼容性，后续更多测试流程是至关重要的。

5.3 研究展望

本文基于 CSRT 算法所改进的算法均基于提高算法运行速度的基准，在未来的研究中随着硬件算力的提升，可以实现该算法与神经网络算法相结合的更进一步的融合算法[17]，在确保算法具备极高的实时性的同时进一步提升算法的可靠性以及对于特定数据集的检测效率，并实现部署于无人机的更为高效率的低延时目标检测系统。

参考文献

[1]. 董红祥. 无人机驾驶员就业景气现状分析报告[J]. 中国人力资源社会保障, 2019, No.116(10): 31-33.

[2]. 冯宏. 基于摄像头的旋翼无人机在热电厂巡检中的应用[J]. 科技创新与应用, 2018, No.243(23): 176-177.

[3]. 刘晓荣. 视频流中基于特征匹配的人体肘关节跟踪算法研究[J]. 电子世界, 2014, No.453(15): 74-75.

[4]. 张代浩. 机场视频监控中基于深度学习的目标跟踪算法研究[D]. 南京航空航天大学, 2019, DOI:10.27239/d.cnki.gnhhu.2019.000299.

[5]. 刘嘉程. 单目标在线视频跟踪算法研究[D]. 山东科技大学, 2020, DOI:10.27275/d.cnki.gsdku.2020.001415.

[6]. 罗泽. 微型无人机图像传输系统的研究与设计[D].中南林业科技大学, 2016.

[7]. 高晨,翟优,郭希维. 融合深度信息的判别式相关滤波跟踪[J]. 物联网技术, 2022,12(01):22-26.DOI:10.16667/j.issn.2095-1302.2022.01.007.

[8]. Henriques, J. F., Caseiro, R., Martins, P., & Batista, J. High-speed tracking with kernelized correlation filters[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, (2015), 37(3), 583-596.

[9]. David S. Bolme, J. Ross Beveridge, Bruce A. Draper, Yui Man Lui. Visual object tracking using adaptive correlation filters[J]. 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2010, DOI: 10.1109/CVPR.2010.5539960

[10]. Redmon J ,Divvala K S ,Girshick B R , et al. You Only Look Once: Unified, Real-Time Object Detection.[J]. CoRR, 2015, abs/1506.02640.

[11]. C. Fu, B. Li, F. Ding, F. Lin and G. Lu, "Correlation Filters for Unmanned Aerial Vehicle-Based Aerial Tracking: A Review and Experimental Evaluation," in *IEEE Geoscience and Remote Sensing Magazine*, vol. 10, no. 1, pp. 125-160, March 2022, doi: 10.1109/MGRS.2021.3072992.

[12]. Redmon J, Farhadi A. YOLO9000: Better, Faster, Stronger[J]. CoRR, 2016, abs/1612.08242.

[13]. Redmon J, Farhadi A. YOLOv3: An Incremental Improvement[J]. ArXiv abs/1804.02767, 2018, n. pag.

[14]. Alexey Bochkovskiy, Chien-Yao Wang, & Hong-Yuan Mark Liao. YOLOv4: Optimal Speed and Accuracy of Object Detection[J]. 2020.

[15]. Chris Hughes and Bernat Puig Camps. YOLOv7: A Deep Dive into the Current State-of-the-Art for Object Detection. Towards Data Science, 2022.

[16]. Danelljan M, Häger G, Khan S F, et al. Discriminative Scale Space Tracking[J]. CoRR, 2016, abs/1609.06141.

[17]. Farkhodov, Khurshedjon et al. Object Tracking using CSRT Tracker and RCNN[J]. Bioimaging (Bristol. Print) , 2020.

致　谢

在这篇论文的完成过程中，我有幸得到了许多人的悉心指导和大力支持，他们的帮助使得我的研究更加完善和丰富。在此，我想向他们致以最衷心的感谢。

首先，我要衷心感谢指导老师范子川老师。您在整个课程设计过程中给予了我无微不至的技术指导和答疑解惑，您的指导使我受益匪浅。您提供的宝贵思路和建议为我的设计提供了坚实的支持，使我能够充分发挥自己的创造力和专业知识。您的教诲将成为我未来道路上的重要指引。

此外，我还要特别感谢申宇老师，感谢您慷慨提供的元器件。您的支持为我完成实践环节提供了关键的物质基础，使得我能够在设计过程中充分验证我的系统，使我能够顺利进行实验和测试。没有您的帮助，我的课程设计将无法如期完成。

在整个过程中，我还要感谢同小组的成员们。你们给予了我巨大的帮助和支持，我们共同协作、相互促进，使我们的项目取得了良好的进展。每一次的讨论和合作都是我成长的宝贵经验，你们的贡献是我无法忽视的。

最后，我要感谢所有给予我鼓励和支持的家人、朋友和同学们。你们的理解、鼓励和支持是我坚持不懈、克服困难的动力源泉。没有你们的陪伴和支持，我将无法完成这个项目。

再次衷心感谢所有在我课程设计过程中给予帮助的人，是你们的支持让我的学习之旅变得更加丰富和有意义。我将永远怀着感激之情铭记于心。

附录

主程序代码：

```
import cv2 as cv
from pynput.keyboard import Key, Controller
from tracker import tracker

keyboard = Controller()

n = 4

cap = cv.VideoCapture("resources/bike.mov")
cap.set(cv.CAP_PROP_BUFFERSIZE, 1)


def selectROI(event, x, y, flags, param):
    global x1, y1, x2, y2, selecting_roi
    if event == cv.EVENT_LBUTTONDOWN:
        print("S")
        x1, y1 = x, y
        selecting_roi = True
    elif event == cv.EVENT_LBUTTONUP:
        keyboard.press('s')
        #keyboard.press('c')    #(Key.space)
        print("C")
        x2, y2 = x, y
        selecting_roi = False
        bbox = [x1//2, y1//2, (x2-x1)//2, (y2-y1)//2]
        tracker.init(frameResize, bbox)
        cv.destroyWindow('Select Target')
    elif event == cv.EVENT_MOUSEMOVE and selecting_roi:
        x3, y3 = x, y
        cv.rectangle(frame, (x1, y1), (x3, y3), (0, 255, 0), 1)



def pressSpace(event, x_, y_, flags, param):
    pass

cv.namedWindow('Select Target', cv.WINDOW_NORMAL)
cv.setMouseCallback('Select Target', selectROI)

while True:
```

```
        ret, frame = cap.read()
        cv.resizeWindow('Select Target', 1920, 1080)
        cv.imshow('Select Target', frame)
        cv.setMouseCallback('Select Target', selectROI)
        selecting_roi = False
        cv.waitKey(1)    # Add a delay here
        frameResize = cv.resize(frame,(int(cap.get(cv.CAP_PROP_FRAME_WIDTH)//2),

int(cap.get(cv.CAP_PROP_FRAME_HEIGHT)//2)),
                                                interpolation=cv.INTER_NEAREST)
        if cv.waitKey(1) & 0xFF == ord("s"):
                break

        '''
        if cv.waitKey(1) & 0xFF == ord("q"):
                break
        elif cv.waitKey(1) & 0xFF == ord("s"):
                bbox = cv.selectROI('Select Target', frameResize, False)
                tracker.init(frameResize, bbox)
                cv.destroyWindow('Select Target')
                break
        '''

while cap.isOpened():
        ret, frame = cap.read()
        frameResize = cv.resize(frame,(int(cap.get(cv.CAP_PROP_FRAME_WIDTH)//2),

int(cap.get(cv.CAP_PROP_FRAME_HEIGHT)//2)),
                                                interpolation=cv.INTER_NEAREST)
        n=n+1
        if n==5:
                success,bbox = tracker.update(frameResize)
                n=0
                if success:
                        x, y, w, h = [int(i) for i in bbox]
        cv.rectangle(frame, (x*2, y*2), ((x + w)*2, (y + h)*2), (0, 255, 0), thickness = 1)
        cv.namedWindow('Tracking', cv.WINDOW_NORMAL)
        cv.resizeWindow('Tracking', 1920, 1080)
        cv.imshow('Tracking', frame)
        if cv.waitKey(1) & 0xFF == ord("q"):
                break

cap.release()
cv.destroyAllWindows()
```