|  |
| --- |
| 中国软件杯--大学生软件设计大赛 |
| 基于计算机视觉的交通场景智能应用设计方案 |

|  |
| --- |
| 团队名称：骑上我心爱的小摩托  队长： 梁止潆  队员1： 李云灏  队员2： 张丁介 |

目录

[摘要 2](#_Toc45548828)

[第一章 系统整体设计 3](#_Toc45548829)

[1.1.系统可行性分析 3](#_Toc45548830)

[1.2.系统需求分析 3](#_Toc45548831)

[1.2.1系统总体需求分析 3](#_Toc45548832)

[1.2.2系统功能性需求 4](#_Toc45548833)

[1.2.3系统非功能性需求 4](#_Toc45548834)

[1.3.系统设计 4](#_Toc45548835)

[1.4.系统创新与特色 6](#_Toc45548836)

[第二章 相关理论与技术 7](#_Toc45548837)

[2.1. Gaussian YOLOv3 目标检测算法 7](#_Toc45548838)

[2.1.1 Darknet框架 7](#_Toc45548839)

[2.1.2 YOLOv3网络结构 8](#_Toc45548840)

[2.1.3 Gaussian YOLOv3网络结构 9](#_Toc45548841)

[2.2. Deep Sort多目标跟踪算法 10](#_Toc45548842)

[2.2.1 Deep Sort 算法核心思想 10](#_Toc45548843)

[2.2.2 Deep Sort跟踪车辆及行人 11](#_Toc45548844)

[2.3.图形界面模块 12](#_Toc45548845)

[第三章 系统实现与测试 13](#_Toc45548846)

[3.1. 开发工具 13](#_Toc45548847)

[3.1.1 模型训练 13](#_Toc45548848)

[3.1.2 检测功能实现 13](#_Toc45548849)

[3.1.3 交互界面 13](#_Toc45548850)

[3.2. 道路以及车辆基本信息的判断 14](#_Toc45548851)

[3.2.1 斑马线的识别 14](#_Toc45548852)

[3.2.2 车道线的识别 14](#_Toc45548853)

[3.2.3 红绿灯的识别 14](#_Toc45548854)

[3.2.4 违停区域的识别 15](#_Toc45548855)

[3.2.5 车牌识别 15](#_Toc45548856)

[3.2.6 车速计算 16](#_Toc45548857)

[3.2.7 车流量统计 16](#_Toc45548858)

[3.3.违规信息的判断 17](#_Toc45548859)

[3.3.1 车辆违规变道 17](#_Toc45548860)

[3.3.2 车辆闯红灯 17](#_Toc45548861)

[3.3.3 车辆不礼让行人 17](#_Toc45548862)

[3.3.4 车辆逆行 18](#_Toc45548863)

[3.3.5 车辆违停 18](#_Toc45548864)

[第四章 结语 18](#_Toc45548865)

# 摘要

随着信息处理技术，特别是深度学习、计算机视觉技术突飞猛进的发展，基于视频分析的智能交通监控系统逐渐成为主流，由传统的人工监视转化为智能化监控对交通道路上的违规行为进行精准识别判断。应赛题组要求，我们设计的智能交通管理系统采用AI、计算机视觉等技术对交通路口场景进行实时监控分析，对出现违规行为的车辆都能准确检测识别，并且能够提取对应监控视频中道路以及车辆的基本信息，实现了多种交通场景应用。

本系统通过道路监控视频收集数据集，使用darknet框架下Gaussian YOLOv3目标检测算法对道路上的车辆以及行人进行检测，获取车辆和行人的坐标信息，并使用Deep Sort多目标跟踪算法进行跟踪，返回识别到的车辆和行人的唯一跟踪编号，以进行后续的交通违规逻辑处理。使用跨平台[计算机视觉](https://baike.baidu.com/item/%E8%AE%A1%E7%AE%97%E6%9C%BA%E8%A7%86%E8%A7%89/2803351)和[机器学习](https://baike.baidu.com/item/%E6%9C%BA%E5%99%A8%E5%AD%A6%E4%B9%A0/217599)软件库 OpenCV的python接口提取道路、车辆的基本信息以及判断车辆违规行为，并对违规车辆进行跟踪拍照并保存照片，方便交管部门进行后续处理。系统测试表明，交通路口监控场景下，能正确提取道路及车辆基本信息，能准确识别检测车辆违规行为，准确率可达90%，检测速率为20fps，达到实时检测的效果。

# 系统整体设计

## 1.1.系统可行性分析

该系统使用python语言进行开发,并使用强大的图形程式框架Qt5的python接口PySide2进行界面开发,该框架简单好用、灵活且稳定性高,使系统开发更为高效,性能更加稳定。在车辆及行人的检测方面,使用了darknet框架进行对模型进行训练预测,该框架灵活、轻型,可以更为方便地从底层对其进行改进与扩展。在提取道路、车辆的基本信息以及判断车辆违规行为方面，使用跨平台[计算机视觉](https://baike.baidu.com/item/%E8%AE%A1%E7%AE%97%E6%9C%BA%E8%A7%86%E8%A7%89/2803351)和[机器学习](https://baike.baidu.com/item/%E6%9C%BA%E5%99%A8%E5%AD%A6%E4%B9%A0/217599)软件库 OpenCV的python接口对图像进行处理。对于用户使用方面,该系统可以部署到上位机,操作简单、界面简洁明了且功能性较强,用户无需具有较高的计算机基础便可操作。

## 1.2.系统需求分析

### 1.2.1系统总体需求分析

近年来，中国经济高速发展，人们收入水平不断增长，汽车开始走进千家万户，并且保持着稳定的增长趋势，给道路交通带来了巨大的压力，道路拥堵率越来越高，对人们出行带来了巨大的影响。同时各类违法违章行为不仅给人们生命财产安全带来了重大隐患，同时也给交管部门的工作带来了巨大的压力。而通过增加交管部门工作人员数量也并不是不可行的方式，这会增加交通管理成本，并且通过不断增加工作人员数量的方式也难以满足当前的工作压力，只能通过现代科学技术来增强警力，向科技要警力。为了构建一个有序、通畅的城市道路交通环境，打击犯罪，公安交通相关部门需要引入更先进的科学技术构建一个智能化、自动化、信息化的智能交通监管系统，旨在尽可能减少交通事故的发生并服务于人民群众的出行，并能够为交通监管者提供更加智能化的监控管理手段。

作为十三五规划的战略高新产业，智能交通拥有很好的市场前景。据市场需求统计预测，到2021年智能交通产业的市场总规模将达880亿元。在高成本投入的同时，智能交通所承担的责任重大，智能交通产业如何结合利用多种高新技术，如计算机技术、物联网、大数据、视频监控图像处理等，实现对场面的监视、控制、引导、安防，仍需要不断探索。未来15-30年交通产业将发生颠覆性改变，新型基础设施和物联网普及等趋势，未来交通系统将是一个万物互联、多源传感、计算机网络等技术集成的复杂系统。

### 1.2.2系统功能性需求

首先对系统的实际应用场景进行分析,该系统主要的应用场景为交通道路上的十字路口系统用户主要为交通管理人员。由此分析并总结出本系统的主要功能需求有以下几点：

（1）系统通过摄像头或者本地输入视频获取监控区域实时的视频流,然后截取当前帧的图像,供后续检测模块的检测;

（2）系统通过视频流提取道路斑马线、车道线、违停区域以及车辆流量、车速、车牌号等基本信息。

（3）系统能够对监控区域中出现的逆行、违停、闯红灯等车辆违规行为进行识别判断，

### 1.2.3系统非功能性需求

经过分析赛题组要求,该系统应具有以下非功能性需求：

（1）实时性需求：为了保证能及时发现车辆违规行为,减少安全隐患,系统应该具有实时性,其中实时性主要分为两方面。首先,系统应在对视频流的处理、分析和检测的过程尽可能的迅速,减少延迟保证系统的实时性。其次,系统应该在检测出有违规车辆时,可以迅速地得到对应的车辆信息；

（2）可靠性需求：为了能最大程度地保证交通道路上的安全性,系统应该具有可靠性。可靠性主要包括两部分,第一部分是检测的准确性方面,系统应保证能准确地检测出违规车辆,尽可能少的出现误检和漏检等情况。第二部分是车辆信息不会出现误报、漏报等情况；

（3）易用性和稳定性需求：考虑到该系统实际的使用场景和使用人员,系统的界面应简洁直观,功能操作简单易懂。同时系统应保证其运行的稳定性,不会出现运行异常,要具有一定的容错性。

## 1.3.系统设计

本系统分为读入视频模块、检测识别模块、跟踪模块、信息判断模块以及信息显示保存模块。采用了Python多进程模型，一个主进程，三个从进程。进程间不共享数据，只通过消息管道进行通信，加快了系统运行的速度，发挥计算机并行处理的长处，减轻了CPU单核的计算压力。

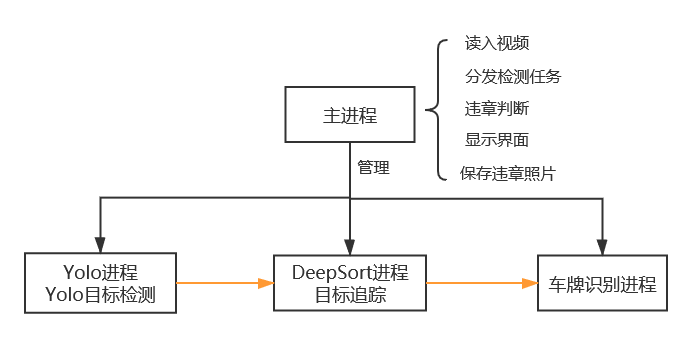


图 1 系统架构图

主进程负责视频读入、各个进程任务的分发与通信、违章判断、显示界面和保存违章照片。每个从进程各司其职，只完成一项单一的工作，充分发挥计算机多核运算性能。

主进程读取视频流后截取每一帧图像输入到检测识别模块，该模块检测道路上的车辆以及行人，并且识别道路上的车道线、斑马线、违停区域等基本道路信息以便于后续模块的判断。

检测到的车辆以及行人输入到跟踪模块，得到各自对应的唯一的跟踪编号，并画出对应的跟踪轨迹。

如果图像中识别到了车辆，就根据车辆位置截取图像ROI输入到车牌识别模块中。保证一个ROI只有一个车牌，提高车牌识别的准确率。

以上模块均为信息判断模块以及信息保存模块的基础，信息判断模块中一部分为对车流量进行统计、对车速进行计算以及对车牌进行识别，另一部分为判断是否出现违规车辆，例如违规变道、闯红灯、不礼让行人、逆行、违停的车辆，如果出现了违规车辆，将违规信息显示在交互界面上，并且保存车辆图片，对于一些基本的车辆信息也会显示在交互界面上，以便客户更好更直观地得到需要的信息。

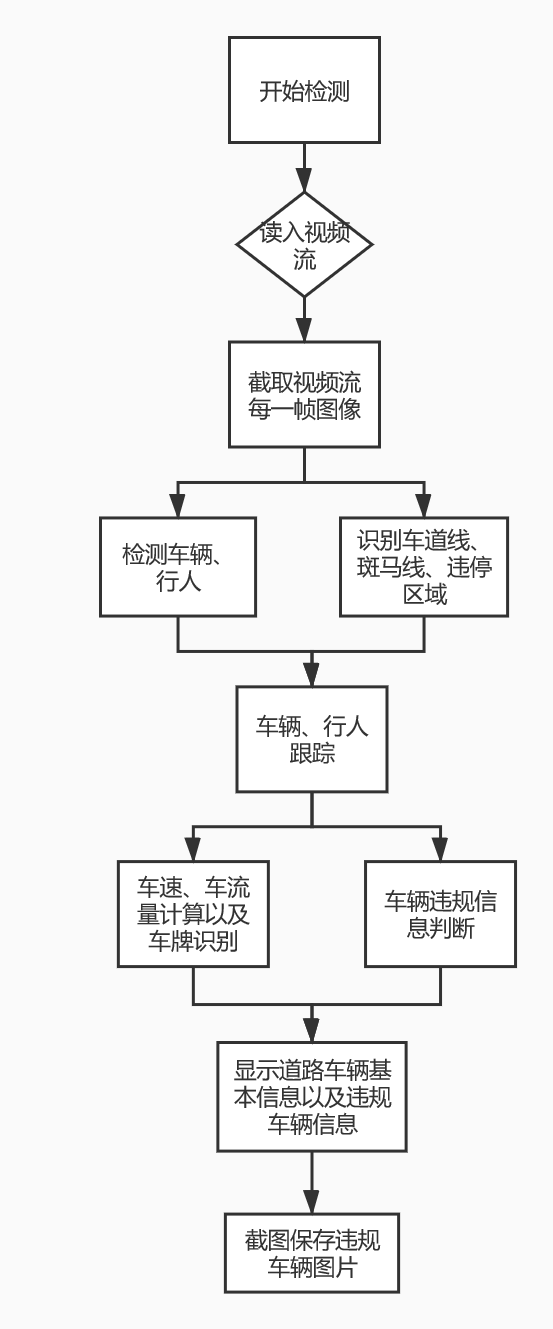


图 2 系统流程图

## 1.4.系统创新与特色

该系统具有以下创新与特色：

1. 灵活性、应用性强，能很好地适应各种复杂的交通十字路口场景。
2. 车辆行人的检测准确率较高，车牌识别以及道路信息车辆违规信息的判断基本准确。
3. 能够在不损失检测精度的基础上达到实时检测的效果。
4. 多进程工作，提高计算机性能利用率。
5. 系统具有友好的交互界面，客户能够清晰的从交互界面上得到车辆道路的基本信息以及车辆违规的信息。

# 相关理论与技术

## Gaussian YOLOv3 目标检测算法

### 2.1.1 Darknet框架

Darknet是一个较为轻型的完全基于C与CUDA的开源深度学习框架,其主要特点就是容易安装,没有任何依赖项,移植性非常好,支持CPU与GPU两种计算方式。

该框架的优点：

（1）易于安装：在Makefile里面选择自己需要的附加项（CUDA、CUDNN、OpenCV等）直接make即可,几分钟完成安装；

（2）没有任何依赖项：整个框架都用C语言进行编写,可以不依赖任何库；

（3）结构明晰、源代码查看、修改方便：该框架的基础文件都在src文件夹,而定义的一些检测、分类函数则在example文件夹,可根据需要直接对源代码进行查看和修改

（4）友好python接口：虽然Darknet使用c语言进行编写,但是也提供了python的接口,通过python函数,能够使用python直接对训练好的.weights格式的模型进行调用；

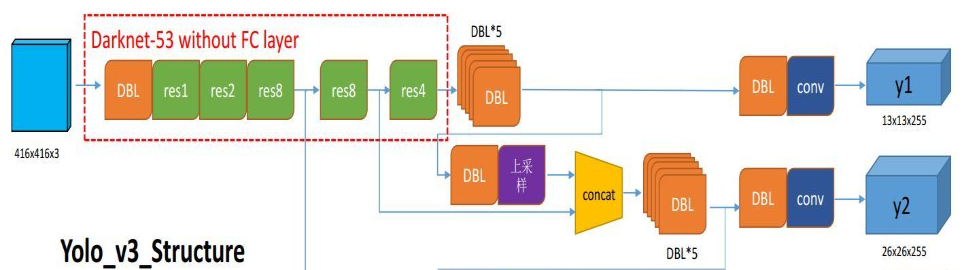
（5）易于移植：该框架部署到本地十分方便,且可以根据机器情况,使用CPU和GPU,特别是检测识别任务的本地端部署,darknet会显得异常方便。



图 3 darknet框架图

### 2.1.2 YOLOv3网络结构

YOLOv3以darknet-53作为基础网络,采用多尺度预测（类FPN）的方法,分别在大小为13x13、26x26、52x52的特征图上进行预测,多尺度预测和特征融合提高了小目标的识别能力,从而提升整个网络的性能。YOLOv3整个网络只由一些卷积层、激活层、批标准化层构成,对于一张指定的输入图像,首先通过darknet-53基础网络进行特征的提取和张量的相加,之后在得到不同尺度的特征图上继续进行卷积操作,通过上采样层与前一层得到的特征图进行张量的拼接,再经过一系列卷积操作之后,在不同特征图上进行目标检测和位置回归,最后通过YOLO检测层进行坐标和类别结果的输出。



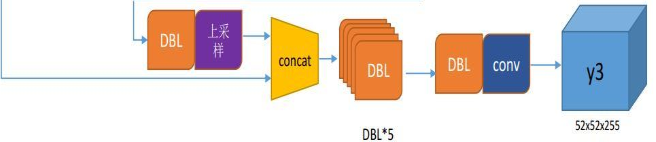


图 4 YOLOv3框架

### 2.1.3 Gaussian YOLOv3网络结构

Gaussian YOLOv3将原始的YOLOv3的目标框输出加入高斯模型,通过增加网络的输出，和改进网络的损失函数，实现了对预测框可靠性的输出。与原始的YOLOv3在坐标预测时输出4个维度不同，Gaussian YOLOv3在bounding box的坐标预测输出中包含了8个维度。这八个维度相当于是预测框中心坐标和长宽，以及对应预测框的不确定性。原作者将这些值建模为四个高斯分布，目标框的位置作为高斯分布的均值，对应的不确定性作为方差。Gaussian YOLOv3通过预测每个坐标位置的不确定性，从而提升最终预测prob值的精确性。由于Gaussian YOLOv3的输出进行了调整，与之对应的损失函数的计算也会做相应的调整。与原始的YOLOv3相比，仅仅调整了预测框坐标位置的回归策略。原始的YOLOv3进行box回归时，由于网络预测输出就是坐标本身，因此计算梯度时就利用了均方误差的方式。而由于Gaussian YOLOv3输出的是均值和方差，因此在计算梯度时就结合了高斯分布的策略。

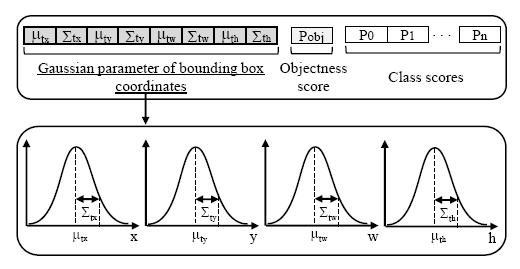
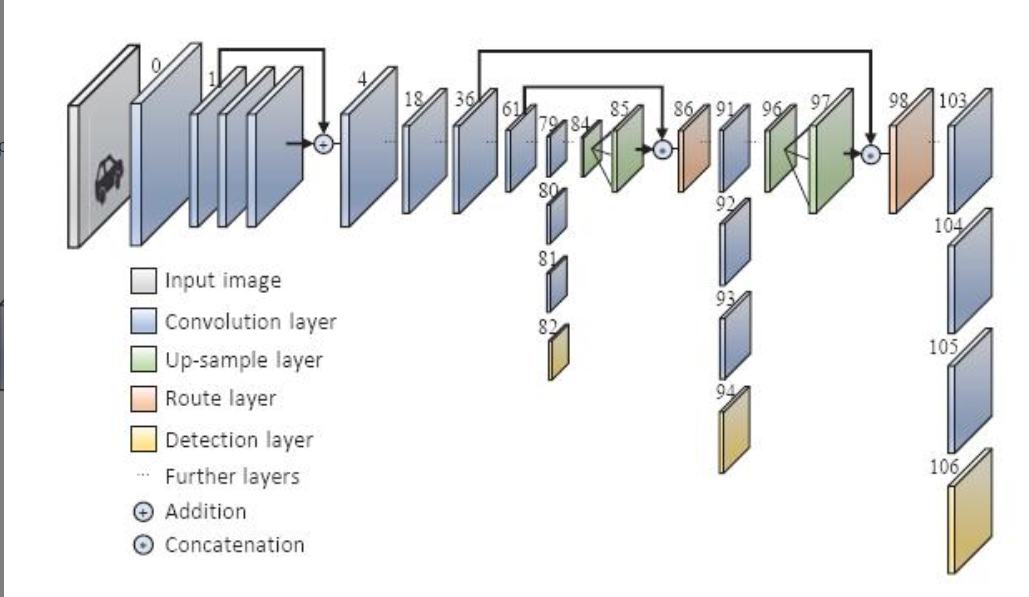


图 5 Gaussian YOLOv3网络结构图

## Deep Sort多目标跟踪算法

### Deep Sort 算法核心思想

DeepSort的跟踪思路为：detection + track 。 所以代码也以检测结果为输入：bounding box、confidence、feature 。confidence主要用于进行一部分的检测框的筛选；bounding box与feature（ReID）用于后面与跟踪器的match计算；首先是预测模块，会对跟踪器使用卡尔曼滤波器进行预测。使用的是卡尔曼滤波器的匀速运动和线性观测模型（意味着只有四个量且在初始化时会使用检测器进行恒值初始化）。其次是更新模块，其中包括匹配，跟踪器更新与特征集更新。在更新模块的部分，根本的方法还是使用IOU来进行匈牙利算法的匹配，只不过对于数据顺序有一些自己的处理：

* 使用级联匹配算法：当一个目标长时间被遮挡之后，kalman滤波预测的不确定性就会大大增加，状态空间内的可观察性就会大大降低。所以针对每一个检测器都会分配一个跟踪器，每个跟踪器会设定一个time\_since\_update参数。如果跟踪器完成匹配并进行更新，那么参数会重置为0，否则就会+1。实际上，级联匹配就是不同优先级的匹配。在级联匹配中，会根据这个参数来对跟踪器分先后顺序，参数小的先来匹配，参数大的后匹配。也就是给上一帧最先匹配的跟踪器高的优先权，给好几帧都没匹配上的跟踪器降低优先权（慢慢放弃）。
* 添加深度学习特征：deepsort在对于sort的改进中加入了一个深度学习的特征提取网络，将所有confirmed的跟踪器（其中一个状态）每次完成匹配对应的detection的feature map存储进一个list。存储的数量作者使用budget超参数（100帧）进行限制。从而我们在每次匹配之后都会更新这个feature map的list，比如去除掉一些已经出镜头的目标的特征集，保留最新的特征将老的特征pop掉等等。这个特征集在进行余弦距离计算的时候将会发挥作用。实际上，在当前帧，会计算第i个物体跟踪的所有Feature向量和第j个物体检测之间的最小余弦距离。
* IOU与匈牙利算法匹配：实际上匈牙利算法可以理解成“尽量多”的一种思路，比如说A检测器可以和a，c跟踪器完成匹配（与a匹配置信度更高），但是B检测器只能和a跟踪器完成匹配。那在算法中，就会让A与c完成匹配，B与a完成匹配，而降低对于置信度的考虑。

还有一些超参数，这些超参数或进行阈值的作用，或进行循环次数的限定，不断帮助算法完成最优匹配与跟踪器更新。之后，将未匹配的跟踪器delete，将未匹配的检测器初始化，将匹配的跟踪器使用对应的检测器进行赋值，作为输出，进入下次循环。

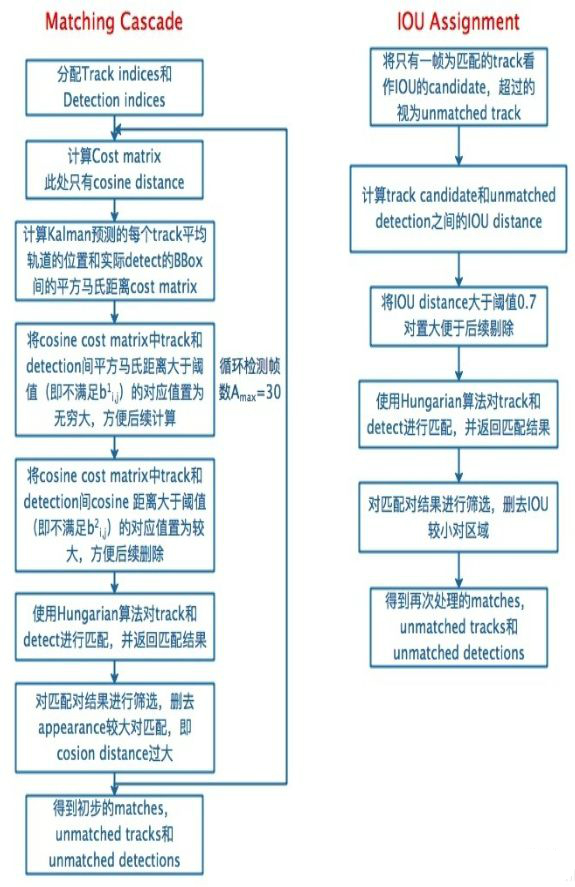
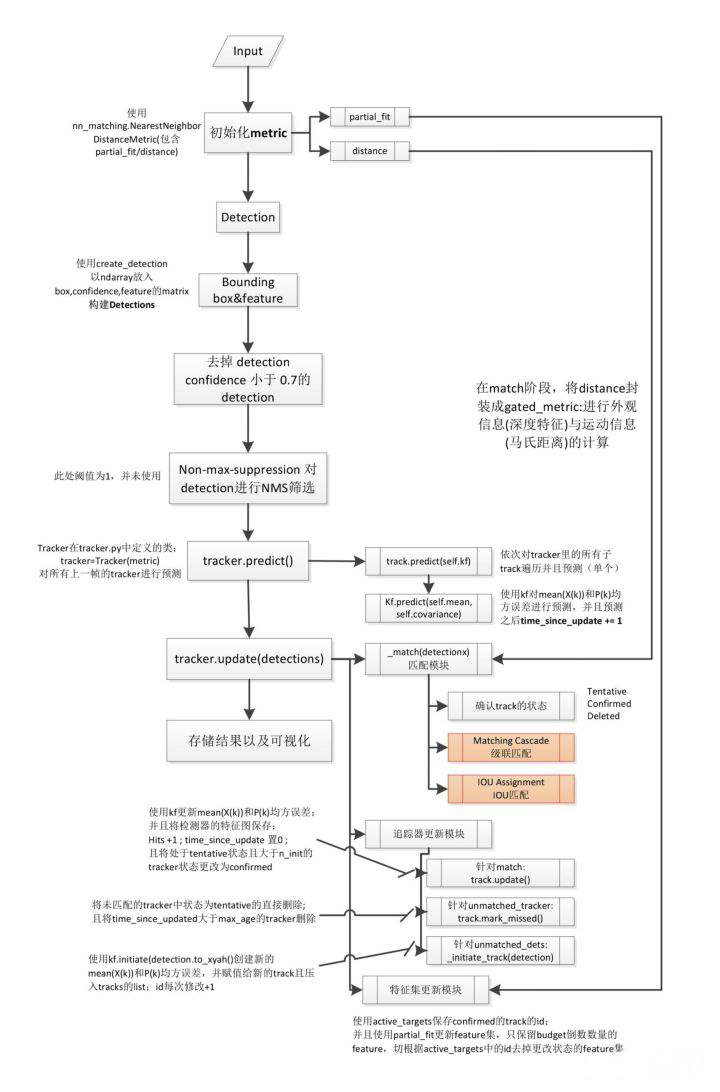


图 6 Deep Sort 框架流程图

### Deep Sort跟踪车辆及行人

本系统只使用Deep Sort跟踪路口的车辆及行人，视频的每一帧从YOLO中获取车俩和行人的坐标信息，输入跟踪器，最终返回识别到的车辆和行人的唯一跟踪编号。根据此编号画出车辆的运动轨迹，并进行后续的交通违规逻辑处理。

## 2.3.图形界面模块

本系统采用PySide2开发图形界面，用于显示Yolo、OpenCV、TensorFlow处理之后的图片，和显示一些必要的提示信息。

PySide2是一个Python模块,它提供了对Qt5.12+完整框架的访问。PySide2作为开源的框架Qt的Python接口既能开发出完美的界面,又能享受Python的便捷开发,并且使用不受限制；与Qt的C++相比,使用PySide2能大大减少代码量。PySide2的这些优点,很适合团队规模不大,项目需要快速推进,需开发精美界面的场合。

本系统设计的图形界面分为了三个部分，分别为基本信息展示区、违规信息警告区和输入视频展示区。

输入视频展示区可以实时展示交通路口摄像头拍摄的视频，经过本系统的处理之后框出视频中的必要信息然后显示在图形界面上。

基本信息展示区可以打印出当前路口的一些基本信息，如车流量、车辆数、系统运行情况、系统日志等。

违规信息警告区会打印出当前路口的车辆和行人违规信息，如车辆违停、车辆违规变道、车辆闯红灯、车辆不礼让行人、车辆逆行等信息。并自动对违规的车辆进行跟踪拍照并保存照片，方便交警进行后续处理。

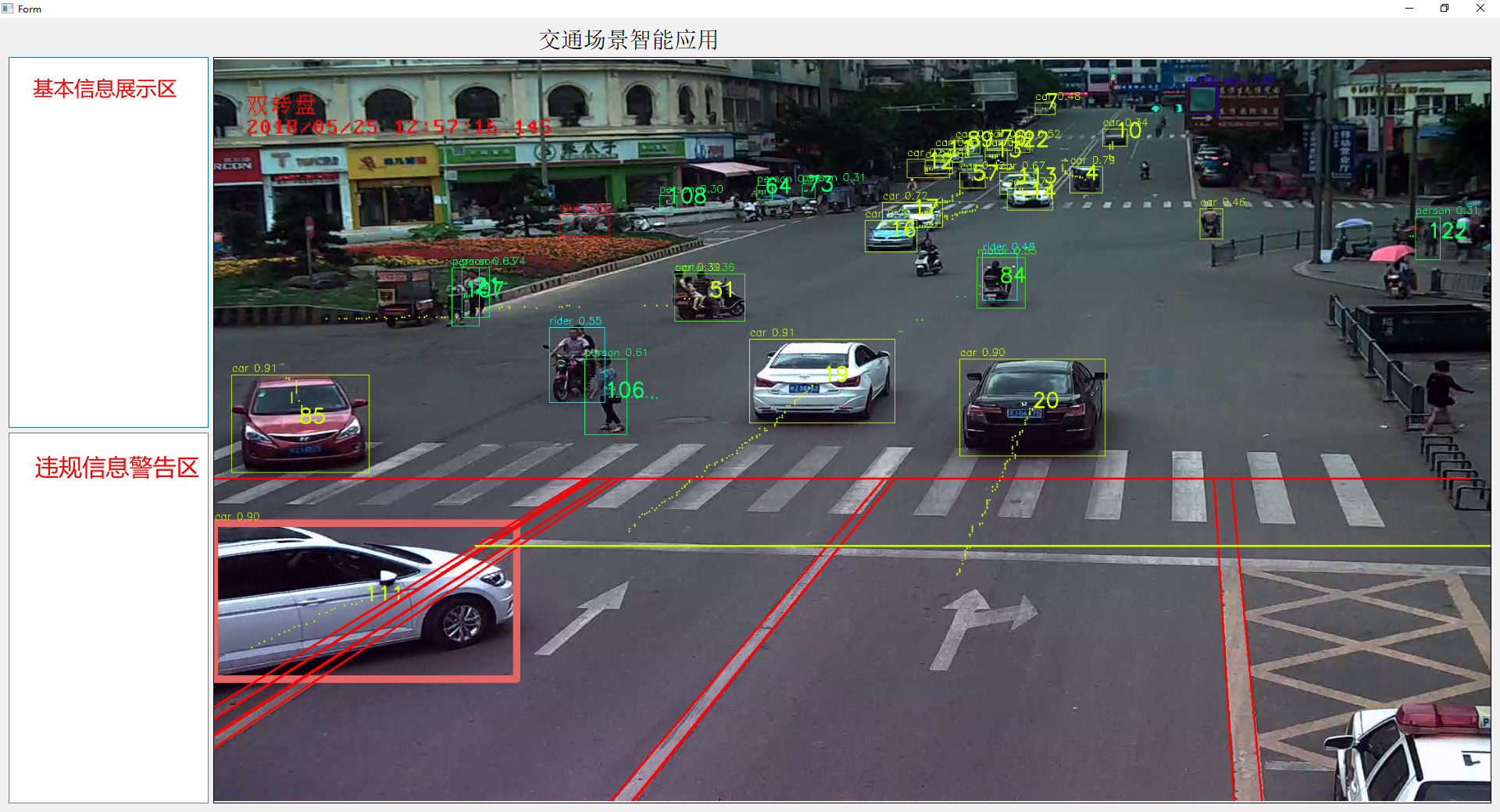


图 7 系统交互图形界面

# 第三章 系统实现与测试

## 3.1. 开发工具

### 模型训练

为了尽可能的提高本系统的运行速度，尽量达到实时检测路口信息。使用C语言开发的darknet框架进行神经网络的训练和测试，并编译YOLO动态链接库便于后续Python调用。经过测试，使用DLL调用YOLO模型会比Python调用YOLO模型快至少200%。

此功能使用CUDA、CUDNN、VS2019等开发。

### 检测功能实现

每一帧经过YOLO识别后返回识别到的物体坐标点列表，再把此列表输入跟踪器得到车辆和行人的具体跟踪编号。然后使用OpenCV实现道路及车辆的基本信息判断，最终把必要的信息显示在交互界面上。

此功能使用PyCharm、OpenCV-Python、TensorFlow2.1等开发。

### 交互界面

交互界面使用PySide2模块进行开发，使用QT设计工具设计本系统的UI框架，然后转化为Python模型并封装到本系统中。

此功能使用PyCharm开发。

## 3.2. 道路以及车辆基本信息的判断

### 斑马线的识别

由于道路上斑马线均为白色，故首先使用HSV空间阈值筛掉图像中大部分别的颜色的区域，然后使用形态学操作腐蚀膨胀减少噪点，填补线磨损增强斑马线的边缘识别效果，让斑马线区域更为规整，canny边缘检测得到基本为长方形的斑马线区域，并寻找图像中的轮廓，使用面积筛掉小面积的轮廓区域，单条斑马线的长宽比例基本确定，使用外接矩形框的长宽比例筛掉大部分轮廓，然后对边缘检测图用sobel求横纵梯度，得到梯度的模值和方向（方向归为0-90°），先过滤梯度太小的点，在滑窗内统计0:80°间隔5°的点直方图，取峰值方向的点数作为判断量，高于阈值判为斑马线。

### 车道线的识别

车道线检测主要使用霍夫变换，再通过进一步处理得到。首先对图像进行灰度转换，在灰度图里面进行找轮廓，之后对图片进行高斯模糊，再进行边缘检测，通过调整参数尽量使边缘变得清晰，但又不会有太多干扰。之后进行ROI感兴趣区域的截取，因为我们的检测目标主要为图片的下方车道线，因此我们将ROI区域放在下面车道线位置，这样就排除了不感兴趣区域的干扰，最后对ROI区域的边缘进行霍夫变换，通过调整参数找出区域内符合条件的直线，为了便于实现之后的功能，我们统一将所有车道线延长至斑马线的中线。

### 红绿灯的识别

截取目标检测得到的红绿灯区域，并对其进行颜色识别。首先使用HSV空间中红、黄、绿三种颜色阈值去除背景部分，并分别统计三种颜色中的非零像素点数，取其最大值，若最大值为红色区域，则判断为红灯，最大值为绿色区域，则判断为绿灯，剩下的则为黄灯。

### 违停区域的识别

所谓违停区域即指地上标注着黄色网状线的区域，提取区域的特征可以知道，我们主要根据检测出的直线位置关系及符合位置关系的直线的数量在进行判断。首先，我们应明确违停区域在理论上的位置：一般出现在最靠边的车道，因此我们只需截取最靠边的车道的ROI区域即可，而分车道的方法主要是对所有车道线进行分组，并通过位置关系确定识别到的每一个车道的信息。因此，我们截取右下角车道ROI进行处理，处理的方法和车道线大同小异，

也是运用霍夫变换，通过调整出来的直线的长度和角度进行筛选，最后获得需要的直线。之后，我们再利用这些直线，判断互相平行的直线的个数，通过个数来判断这个区域是否为违停区域，如果个数超过最大限定，则我们认为违停区域存在，而所谓违停区域即指该车道两边车道和停车线图片底线包围起来的区域；如果没有超过最大限定，则我们认为图像内不存在违停区域。

### 车牌识别

车牌识别采用开源的基于深度学习的高性能中文车牌识别库HyperLPR，并进行封装和简化。首先根据识别到的车对原图截取ROI，舍弃远处过小的车辆，只对近处有可能识别出车牌的车辆进行车牌识别。

得到车辆的ROI之后，首先对车辆图片进行左右边界回归，识别出可能的车牌具体位置，得到车牌位置之后进行图片分割，分割出独立的数字区域，最后使用基于GRU序列的OCR模型来识别单个字符区域的字符类型。拼接在一起就得到了最终的车牌号。

经过测试，在车辆大小适中、车牌号清晰的情况下，车牌识别的准确率能达到90%。足以辅助交警进行车辆违规的判断。



图 8 车牌识别示例

### 车速计算

由于有了跟踪器的加持，对车辆速度的计算就容易很多。首先先确定摄像头在道路的位置信息作为标定。获得比例尺之后，对跟踪成功的车辆计算连续两帧的摄像头像素上的速度信息，通过相机标定映射成真实环境下的车辆速度。但是在系统开发的时候我们并不能对相机进行位置信息的标定，故只计算出了车辆的像素速度，并依据此进行车辆的超速判断。

### 车流量统计

我们规定车流量10s一个周期计数，单位为辆每分钟，规定画面中线为参考线，来对两个方向（不考虑左右方向）来往的车辆进行计数，我们根据之前求出的每个车辆的轨迹，判断是否有相邻两帧的位移连线与参考线相交，如果相交则判定车辆经过参考线，据此计数，每个周期计数汇总输出并清零进入下一个周期重新计数，对10s周期计数换算成每分钟的计数即为检测的车流量。

## 3.3.违规信息的判断

### 车辆违规变道

根据交通规则，机动车不能压过实线车道线进行变道操作，根据这个规则我们需要对违规变道进行判断。首先我们需要检测车道线，车道线的识别在前面已经详细讲解，在这里不做过多阐述。检测出车道线以后，我们根据每辆车的运动轨迹，判断是否有相邻两帧位移连线与车道线相交，如果与车道线相交，我们认定该车为违规变道，记录下该车的信息。

### 车辆闯红灯

在红灯的情况下，画出每辆汽车上一帧和当前帧的中点坐标轨迹，判断该轨迹是否与车辆停止线相交，若相交，则将其列为疑似闯红灯车辆，并跟踪该车辆，若该车辆在还未通过路口中央的时候停住了，则将其从疑似列表中剔除，若该车辆继续运动并通过路口中央，则将其判定为闯红灯车辆，并进行拍照保存到本地，同时识别出该车辆的车牌号，以便交管部门进行统计处罚。

### 车辆不礼让行人

在该功能中有两个元素：车和人，两个都要进行分析。同时，我们判断不礼让行人的流程是：车辆经过行人将要经过的路线，车辆经过后，行人经过车辆通过的路线，此时，我们判定该车辆没有礼让行人，这两个条件任意一个不满足，我们都无法判定车辆不礼让行人。首先，我们根据得出的行人运动轨迹，对相邻几帧的位移连线进行延长，得出行人预判走出的路线，然后根据车的运动轨迹，我们判断是否有相邻两帧位移与行人预判路线有交点，如果有交点，则我们判断该车经过行人将要经过的路线，第一个判断因素达成。之后，我们根据车的轨迹解出车的经过路线，如果行人的相邻运动轨迹与该路线有交点，则我们判断行人走过车的运动路线，第二个判断因素达成。两个判断因素达成，则我们认定该车辆存在不礼让行人的行为，记录该车辆的信息。

### 车辆逆行

根据实际应用条件，我们认定每个路口有一个摄像头，每个摄像头只需检测自己一边的车辆逆行情况即可，因此，我们的车辆逆行检测之针对于上下方向并且靠近摄像头一方的方向。利用之前的车道线检测，我们根据最左边的车道线延长，做检测区域的左端，认定图像距离上边缘三分之一的位置为检测区域上端，认定图像距离右边缘八分之一的位置为检测区域的右端，认定下边缘线为检测区域的下端，这些线构成我们的检测区域。由于摄像头角度的原因，我们不能简单根据车辆的运动速度x、y分量来判断车辆是上行还是下行，因此我们重构参考坐标系，我们以平行于最左端车道线向上的方向为y方向，原x方向为新x方向。定好坐标系后，我们把一段时间内车的位移向新的x、y方向进行投影，判断x、y方向哪个方向运动位移最大，y最大则我们认为是上下行驶的车辆，之后我们再判断车的y分量是正还是负，如果是正，说明车正在向画面的上方行驶，即顺行方向；如果为负，说明车在向画面的下方行驶，即逆行方向，此时我们认定该车存在逆行行为，记录该车的信息。

### 车辆违停

对于车辆违停，我们需要先判断是否存在违停区域，根据上面的违停区域检测，如果没有检测出来违停区域，我们认为现在没有违停车辆。如果我们检测到了违停区域，我们需要判断是否有车辆在区域范围内违停，此时，我们根据画面上的车的轨迹，通过每个车当前坐标判断，车是否在违停区域，如果没在，则不存在违停。如果存在，则我们还需要看该车近十帧是否静止，如果静止，则我们判断该车违停，并记录出测量信息。对于判断静止，我们认为只认定近十帧位移不变不准确，因此我们判断十帧之内的平均差不大于某个值就认为这辆车静止。

# 第四章 结语

随着时代的不断进步，技术的不断发展，基于视频分析的智能交通监控系统逐渐成为主流，由传统的人工监视转化为智能化监控对交通道路上的违规行为进行精准识别判断。本团队基于深度学习以及计算机视觉技术设计了一套交通场景智能应用系统，对交通路口场景进行实时监控分析，对出现违规行为的车辆都能准确检测识别，并且能够提取对应监控视频中道路以及车辆的基本信息，实现了多种交通场景应用。