基于霍夫变换的家庭入侵检测系统设计

摘要：家庭是重要的私人居住场所，而家庭入侵严重威胁了家人的财产和生命安全，有必要在家庭的出口处设置监控，在暴力开锁之类的事件发生时通知用户并保存监控录像。本系统设计采用了高斯滤波、Canny 边缘检测、霍夫直线变换，检测出待测范围内的直线边缘，然后分析并识别出预先粘贴在门上的矩形标记，进而捕捉到门的活动情况，其中还利用了标记一帧位移有限而干扰矩形产生随机的特性，有效过滤掉干扰的识别结果。由于门打开后标记矩形的角度会发生变化，算法保留了适量的角度、长度、边缘线段断裂等误差，大大提高了检出成功率。系统将事件分为了几种类型：入侵、回家、出门、复合，在第一次门打开时开始记录，门最终关闭或超时后结束。事件开始之前，当标记被遮挡（存在漏洞）或图像边缘框像素区域变化过大时，系统分析认为有室内行人经过，后续开门事件记为出门。门若直接被打开，系统将启动人脸识别，将检测到的人脸上传到 Face++服务器，与已授权身份对比，进而判断入侵或回家事件，超时则默认为入侵。程序有单独的命令行线程，用于人机交互，可以对各个模块内容进行查看和设置。

系统搭建完成后，在几个不同场景下对其进行了门监测模块测试和事件测试，门关闭和打开时都能很好地识别出矩形标记，不同的事件发生时，系统也能立即识别出来，然后提示用户、保存视频录像。

关键词：入侵检测；监控系统；图像分析；Canny 边缘检测；Hough 直线变换；矩形识别

Design of Home intrusion detection system based on Hough transformation

Abstract: Home is an important private place to live, home intrusion seriously threatens the safety of our property and life. It is necessary to set up a monitor at the exit of the home, when an intrusion event occurs such as open the lock violently, system will notify the user and save the surveillance video. This system design has used the Gaussian filter, Canny edge detection and Hough line transformation, to detect the line edges within the range to be monitored, and then analyzes and recognizes the rectangular mark pasted on the door in advance, and then it will capture the movement of the door. The characteristic that mark's movement of one frame is limited in a little range but the interferential rectangles' generation is random is used to effectively filter out the interferential recognition results. Since the angle of the marked rectangle will change after the door is opened, the algorithm retains an appropriate amount of deviation such as angle, length, and edge line segment breakage, which greatly improves the detection success rate. The system will divide the event into several types: intrusion, return home, going out, and compound. The recording starts when the door is opened for the first time, ends when the door is closed finally or timeout. Before the event starts, when the mark is blocked (there was a bug), or the variance of the pixel areas of the frames of the image' edges is too large, the system will analyze that there is an indoor pedestrian passing by, and the subsequent door opening event will be recorded as leaving home. If the door is opened directly, the system will start face recognition, upload the detected face image to the Face++ server, compares it with authorized identities, and then judges it as the intrusion or return home event. The timeout event will be judged as default intrusion. The procedure has a particular command-line thread for human-computer interaction, you can view or set the content of each module.

After the system was built, the door monitoring module test and event test were conducted in several different scenarios. The rectangle mark can be well recognized when the door is closed or opened. When different events occur, the system can immediately recognize it, and then will prompt the user, and save the video recording.

Key words: family intrusion, image analysis, Canny edge detection, Hough line transformation, rectangle detection

目　　录

第 1 章　绪　论

1.1 研究背景及意义

家庭入侵检测是一个很具有研究价值的领域，因为住宅很容易成为犯罪分子的作案对象，即使在当今法制健全的时代，家庭入侵事件也时常发生，对家庭成员的财产与人身安全有极大威胁。在没有充足准备的情况下，仅依靠事后公安部门的搜查、天眼系统的捕捉技术，效率显得很低，也容易疏漏掉一些行事谨慎、有充分应对措施的作案人员。即使小区的安保系统有一定的安保能力，仍然是不够的。

所以私人住宅最好是自行安装一套监控系统，将摄像头置于住宅出口前，将进出门的所有事件以视频的方式保存下来。单独的监控录像保存的视频太长，用户要定位事件发生的时间则需要完整浏览历史记录，很麻烦，并且不能实时通知用户。所以本设计的目的就基于此，系统能够在入侵或其他事件发生时通知用户，并保存录像。

入侵事件与出入口房门的开关状态有密不可分的关系，捕捉房门的活动，将有助于判断入侵是否发生。特别设计的在门上面粘贴的矩形便签标记，能在门活动时发生移动，于是识别矩形标签的轮廓位置，并与门关闭时的初始位置做比较，就可以捕分析门的状态。识别矩形轮廓的关键是识别四条直线段的边。对图像的每个像素点求出梯度，再处理，就得到一张梯度二值图，也就是边缘检测图。霍夫变换是一种对曲线方程中参数和变量互相转换，将在图像空间检测直线转换成在参数空间中查找极值点的技术，霍夫变换可以高效地检测出边缘检测图中的直线段。这就是边缘直线检测的过程，结果中就有矩形标记的四条边的信息，边信息可以被提取出来并构建标记轮廓矩形，从而识别到标记、检测到门的状态。

本系统的门状态监测，是一个可靠的图像分析算法，比起利用计算机视觉检测目标来判断是否发生入侵，更加稳定。再结合室内遮挡检测、进门者人脸识别与对比，有效地将危险事件与安全事件区分开来，除此之外，本系统还实现了一些有用的功能。如：提醒忘记关门，开门即警告（避免人脸识别漏洞）、标记消失过久警告、远程提醒。设计实现的功能，有效地提高了家庭住宅的安全性，并且方便了家人的出行，对房屋的出入管控智能、高效、安全、稳定。因此，基于霍夫变换的家庭入侵检测系统具有重要的研究和现实意义，及广阔的发展前景。

1.2 国内外研究现状及存在问题

在家庭入侵检测领域，国内外已经有一些可行的方案。比如采用红外热传感器、超声波传感器检测目标、监控录像拍摄视频。对于红外热传感器，如果检测到生物入侵的信号，就向用户发起警告，用户可以远程观看摄像头拍摄的影像。这种方案比较简单和实用，能很准确地检测出生物发出的红外光，十分稳定。但很多智能型的功能，这种方案都是不具备的，如：身份识别、室内行人检测，对出门进门、安全与否做区分，这些功能就不得不用到图像分析处理、计算机视觉等技术了。传感器可以作为升级内容，加在入侵系统里，它在提高入侵识别率方面，是有帮助的。

基于计算机视觉的视频监控领域，在国内外已经有了很多研究，例如：为实现安防系统的自动化、智能化，合肥工业大学研究出了一种基于 OpenCV 的家庭入侵检测嵌入式系统。使用开源的计算机视觉库、方向梯度直方图特征算法和 haar-like 特征算法技术。并且系统采用了相关性的方法和自适应改变阀值的方法，进行补偿控制，这能有效减小因自然光线强度变化而引起误报、漏报概率。这一方案基于机器学习的技术，可以检测过往行人，但因为人工智能算法计算量大，技术不够成熟，容易出现漏洞，受环境和入侵者持有防护措施的影响，仍然有极大的误报、漏报风险。

霍夫变换是一种有效的从图像中提取有表达式曲线的方法，其中霍夫直线变换能以很高的效率从图像中提取出所有的直线或线段，配合边缘检测，组成边缘直线检测函数，这已经在国内外相关研究中成为主要的非机器学习图像目标识别的主要方法，在许多应用设计里发挥了重要作用，如：车道线识别算法、证件图像尺寸恢复技术、斑马线识别、手势识别、四边形分类识别等，这些应用的特点通常是目标特征明显、边缘轮廓清晰。

入侵检测之所以能和霍夫变换结合起来，在我看来，不是因为该技术能直接检测出行人、入侵者这些目标，因为霍夫变换只适合检测直线、圆等简单目标，对于复杂的情况，霍夫变换无能为力，理论上，其可以检测任何表达式的曲线，但是复杂曲线的参数通常较多，这会使变换算法的复杂度急速上升。如果仅使用简单图形构建复杂的行人目标，我认为难度太大，甚至不可能，这些复杂物体的变化形态根本不好用具体的图像分析算法描述，还是应该把任务交给机器学习、分类器或向量机等算法。

可以结合的原因，是我们可以在房屋出入的门上，找到或制造一些简单图案的物体，比如以直线或圆为轮廓的，这些物体在门移动时移动，识别这些物体并加以分

析，就能判断门的状态，结合其他技术，就可以判断入侵了。

1.3 设计目标与实现方法

1.3.1 房屋出入管控

通过分析房门状态、检测是否有室内行人经过，事件首次发生时，判断从室内还是从室外打开。如果从室外打开，还要进行人脸识别和身份对比，将其归类入侵或是回家。在出门或回家事件中，如果门打开时间过长，就提醒忘记关门。因为人脸识别对比可能不稳定，或入侵者伪造图像，身份认证就失去效能了，所以一旦门从室外打开，就要向用户发出警告。一次完整的事件在门第一次打开时开始，最后一次关闭且录制结束后结束，期间再次开门，记为复合事件，事件继续，录制继续。这一步的目的是为了防止门反复开关，事件视频过短和过多。

本设计采用图像分析对房门活动进行分析识别。门监测的关键就是确定检测的目标，该目标的移动、变化应该是门相关活动发生的充分必要条件，其充分性在于：检测到目标活动，就可充分推导出门的活动，确保这一过程是正确有效的；其必要性在于：任何门的活动，都必然导致目标相关活动发生。本设计在完成之前尝试了几个目标：靠近门框的门扇左右侧边缘、门扇顶部边缘，但实验证明这些方案是几乎不可行的，最终我采用了以门扇上粘贴的矩形标记为目标的更有效的办法，矩形标记是一种预先准备好的与周围门扇颜色对比度高的单色长方形纸片，为适应夜间环境可将其更换为颜色鲜明的荧光贴纸。标记移动时，可以推导出门在移动，门移动时，标记也一定跟着移动， 标记的矩形特征是区分与其他物体的第一保障，识别标记的步骤如下：

（1）图像边缘检测处理，其实质是高频滤波，对图像的每个像素点计算梯度，在变化率高的物体边缘处信号量被放大。本系统使用的是 OpenCV 的 canny 算法。

（2）从边沿图像中提取直线段，需要利用到霍夫变换技术，该方法的基本原理是对曲线方程中参数和变量互相转换，原空间中的曲线变量是变化的，参数是固定的，互相转换后的表达式则变成了：参数变化、变量固定，在参数空间中，该表达式显然指的是一个点，于是问题就从寻找原空间中的线转换成寻找参数空间中峰值大的点，图像处理中即为统计每个像素点经过的由原图像像素点转化的曲线数。矩形识别算法可以将各个角度的矩形识别出来，然后利用标记位移连续特性过滤干扰结果。本系统使用的是 OpenCV 的 HoughLinesP 算法。

（3）识别矩形，利用矩形的四边关系，依次进行位置比较、忽略较小缺口，这之前还要修补残缺线段，包括水平和垂直。

（4）从矩形结果中找到标记项，利用了门移动的连续性，标记运动也是连续的，大小也不会发生突变，而室内物体（如：人）产生的干扰矩形通常是随机生成的。虽然门打开后室外很容易出现新的矩形物体，但只要避免与原位置、大小相同，程序就不会产生门已关闭的误报。需要注意的是门关闭时监测范围内不要出现其他矩形。

室内行人检测并不使用计算机视觉技术，而是设计了 2 种特别的方法：

（1）遮挡标记，主人从房门离开前，通常会遮挡住标记，利用这一特性，若标记消失时间过长，就可以认定为有室内行人经过 后续的开门事件视为出门。该方法存在很大漏洞（见 2.4.5）

（2）判断图像外框范围内是否发生剧烈变化，外框是指一定边距以外的“回”字型范围，算法需要该部分始终处在室内，算法的原理是：家庭成员出门前必定进入画面，该范围内的图像会发生剧烈变化。

人脸识别则采用了 OpenCV 的级联分类器 API，人脸对比是将识别到的人脸上传到 Face++云服务器，调用 API 接口，与已授权人的图像或识别码做对比。

1.3.2 实时提醒

事件发生时，除了在命令行中输出信息以外，还要立即向手机发送 QQ 消息，让用户能够及时反应，做出应对措施，如：

（1）当主人在外时，发生了入侵事件，或者有人进屋且身份认证通过，但明确知道该时间段家人都在外面时，就应该通知小区保安，或者报警。

（2）如果家人在家休息，有非法入侵发生，很有可能面临生命安全的风险，如果没有听到开门声，手机依然能收到消息，家人就可以将卧室房门反锁，然后报警。

（3）如果出门或回家一段时间后内门没有被关上，就会向手机发送消息，通知用户及时返回并关闭房门。

（4）门关闭状态下，标记被识别到消失时间过长，就通知用户，可能是环境因素，可能是程序漏洞，总之，用户应该保持警惕，因为这时是监测不到入侵事件的。

向手机发送 QQ 的原理就是：模拟 Windows 的键盘按键，向 QQ 设备窗口“我的

Android 手机”发送英文信息。该系统只设计了桌面软件，暂时并没有设计远端部分，如果今后有机会，可以设计一个在 Android 下的监控客户端 APP，或是在 PC 上的桌面客户端软件，这样，就能提供更多强大的功能，比如实时传输监控画面、对监控系统进行参数设置或其他配置、查看历史监控录像和事件录像等。

1.3.3 案件回溯

首先，系统具有保存历史监控录像的功能，并且可以定时删除过期的视频文件，这样，在案件侦破工作中，即使系统出错没有录下入侵监控视频，也有完整的参考数据，在平时用户也可以了解到房门处发生的一切事情，这保证了系统的数据完整性。

任何事件发生时，都会启动事件录像流程，把从事件开始前一分钟，到事件结束和结束阶段的录制时间超时时的整个过程的图像，以视频方式保存到本地，对于入侵事件，文件放在 exception（异常）目录，对于其他事件，文件放在 event 目录，文件名中还包含了事件类型、门打开超时标记、复合标记、事件发生时间的信息。

事件发生时还会将各种信息保存到 log.txt 日志文件中，其中有：事件开始时间；事件中的每一次开门、关门、超时的时间点；每个子事件的类型、室内行人经过的检测时间、人脸识别的时间、若身份认证成功的人的名字（如：Alice、Bob）等

1.4 主要功能指标

（1）标记单次识别成功率，与环境、标记与门对比度、直线边缘检测各项参数、角度长度偏差、阈值、干扰矩形数量等因素有关；

（2）标记最大识别前失败次数，连续检测失败才被视为标记消失，不得超过连续失败最大值的一半，否则，在遮挡标记行人检测方法下，极有可能出现入侵事件误判为出门事件的情况，一般来说，标记单次成功率越低，该指标就越高；

（3）人脸识别误判率，其与环境、参数设置有关，因为可重复多次识别和对比，就极大减小了识别失败概率；

（4）事件漏判率和误判率，其他所有因素的影响都体现在这两个指标上了，可能的错误情况有：门打开未检测到标记移动、门关闭未检测标记归位、入侵前误判室内遮挡、身份识别错误等。

经实验调试，门关闭时，即使加入干扰物体（不遮挡），标记的单次检测率也能稳

定在 70%以上，门打开后高于 50%，标记最大识别前失败次数通常情况下小于 10，而连续失败最大值通常设置超过 20，人脸识别误判几乎不存在。除开利用程序漏洞的方式，事件漏判和误判也几乎不存在。

第 2 章　系统设计

2.1 总体框架

本系统最终成果是运行于 Windows 下的 PC 桌面软件，并不采用嵌入式的方式，是因为设计的关键在于算法的编写和软件系统的搭建，故不考虑自建硬件，如果以后有具体产品的需求，可以把该系统的核心部分移植到嵌入式系统中。软件采用的语言是 C++，开发环境为 Visual Studio2019，软件的界面简单，是命令行+图像窗口的模式。该程序子部分数量较多，流程比较复杂，所以需要对项目的不同功能部分做严格划分，故采用了面向对象的模块化设计，几乎每个模块都设计了相关主体类和辅助类，因为信号采集与文件储存、标记及门状态监测、人脸识别、命令行等几个部分相对独立，运行时间都较长，于是本系统被设计成了多线程，这加大了程序设计的复杂度，过程中时刻要注意以当前线程为视角考虑问题，公共资源要做好访问权限管理。

本系统的模块有：全局模块、门监测模块、人脸识别模块、事件控制模块。本系统采用多线程编程，主要的有：主线程、命令行线程、门监测线程、人脸识别线程，文件删除和事件视频生成由于耗时也会另起线程，以下是各部分简介：

（1）全局类管理文件名、全局参数，global.h 下定义了一些通用函数；

（2）画面采集与视频文件存储，以 main()函数为入口的主线程独占摄像头的 Video Capture，实时采集摄像头画面，把其他模块的检测结果显示在图像窗口。主线程还负责保存文件工作，保存历史视频，删除过时历史；保存最近 1 分钟和前一分钟的临时文件，事件发生后就可以保存至少 1 分钟前的实际情况；保存事件余下数据，当新临时文件满 1 分钟后，会创建事件余下视频文件，事件结束后，拼接这三个文件（也可能缺少旧临时文件、余下文件），放入 event 或 exception 目录里；

（3）摄像头类管理摄像头资源、当前帧图像、帧计数、摄像头参数；

（4）门监测模块作为最重要的子模块，目标是识别标签，利用其与门状态的关系，向事件控制模块上报门的状态、室内遮挡状态。

（5）人脸识别模块也有单独线程，但平常情况下处于休眠状态，进门事件发生后，唤醒该线程，检测人脸，处理后上传 Face++服务器，与已授权身份对比，成功则将事件状态设置为回家，否则若尝试次数超过 3 次或超时，设置为入侵事件；

（6）事件控制模块利用门监测模块上传的状态，综合分析，得到各事件是否发生的结果，发生后通知用户、通知主线程准备视频文件；

（7）命令行系统，每个模块都有各自的命令行处理接口，命令行线程在接收到用户的 set、param、（模块）等命令后会执行这些接口，也就达到了人机交互的目的。

程序已将大部分主要的类静态实例化了,如 Camera 对象为 camera，DoorMonior 类对象为 door_monitor，FaceDetector 类的对象为 face_detector。

2.2 全局与摄像头模块

Global 类保存了根目录、各子目录和文件的命名，程序初始化时先判断默认根目录是否存在，否则提示用户设置路径，该类还有视频长度、历史文件过时参数设置。

ProcCtrl 类管理程序暂停、启动和重启，stoped 变量向各线程指示程序是否暂停，程序暂停后主线程会将 wait_counter 设为 2，门监测、人脸识别线程清理数据后使其减 1，变量归零后主线程才继续。

Timer 类是一个计时工具，程序中许多地方要对算法计时和一定频率的输出调试，这个工具简化了这一过程，通常 2 秒一次调试输出，不至于命令行被这些内容占满。

to_string 和 to_time_t 函数是时间、字符串转化的便捷工具，采用通用格式，在带时间的调试输出、文件名设置与解析等地方发挥了重要作用。

AnsiToUnicode、UnicodeToAnsi 是字符串编码转换工具，WindowsAPI 许多地方是采用 Unicode 编码。Affirm 函数用于命令行人机交互中的确定选项，有几个版本。

摄像头模块 Camera 有 current 当前帧的图像、帧计数两个重要变量，由一个共享锁控制访问。摄像头参数有：id（设备号）、size（分辨率）、fps（帧率）、brightness（亮度）、contrast（对比度）、saturation（饱和度）、hue（色调）、exposure（曝光度）。

2.3 画面采集与视频文件存储

由主线程全程负责，进入主函数后，首先对目录、各文件名、窗口初始化，再开启命令行、门监测、人脸识别线程，用 cv::VideoWriter 创建第一个历史、临时文件。然后进入主线程循环，首先用“camera.cap >> pic”采集图像，判断程序是否暂停或重启，首次暂停后会清理文件，再通知其他线程，等待其他线程清理完再继续，程序暂停后门监测、人脸识别线程都会属于休眠状态，通常用来设置模块参数的；首次启动

时重新创建历史文件、临时文件；而重启则不会清理历史和临时文件，仅通知子线程清理数据。

非暂停条件下主线程下一步是将采集图像保存到 camera.current，供其他线程使用，将 camera. frame_ counter 帧计数加一，其他线程由此可以计算用时，在许多地方不需要再调用系统定时器。接下来判断文件长度，决定是否发布并创建下一视频文件，其中在没有事件发生时，将临时文件“Old.avi”删除，“New.avi”改为“Old.avi”，有事件发生时将它们改为对应的事件类型文件名，再创建“EventRest.avi”，在事件结束并超过记录时限后设 event_ctrl.status 为 NoEvent，开启新线程组合这些文件，放入对应的文件夹 event/exception，其文件名指明了事件的类型、是否包含至少一次开门、是否开门超时和事件发生的时间。做完这些判断后就向视频文件写入数据。

不论是否暂停，接下来都要把图像显示在窗口上，在这之前要根据门监测、人脸识别的相关参数，决定是否在图像上绘制监测范围、检测结果等线条或矩形框，若模块设置有 is_save,这一过程会在写入视频文件前进行，于是保存的文件也会有这些线条。调用 cv:: waitKey(camera.fps / 6)短暂等待图像的显示。

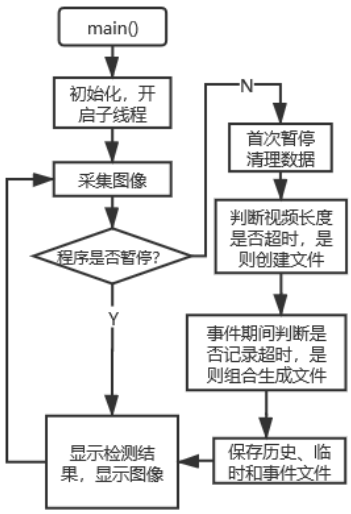


图 2.3 主线程流程框图

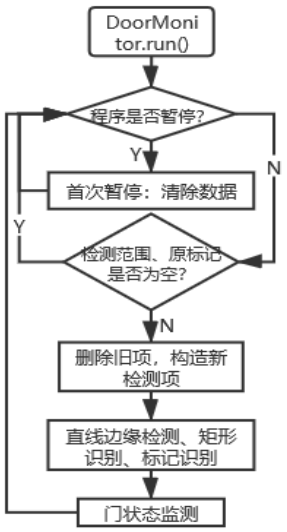


图 2.4.1 门监测线程流程框图

2.4 门状态监测

2.4.1 主要结构

在 door_monitor.h 下定义了三个主要类，LineDetectItem 是用于保存一次直线和矩

形检测数据和图像（原图、预处理、边缘图）信息的，RectDetectItem 是专用于保存某一次矩形检测目标的连续性情况，DoorMonitor 则是最主要的门状态监控器。DoorMonitor 有两个 LineDetectItem 指针对象成员:last_item、current _item，由于标记检测的整个过程中都涉及了对检测数据的读取写入操作，不好做访问权限控制，利用锁机制可能严重阻塞主线程，但主线程的窗口又需要显示检测结果，于是就设计了这两个对象，窗口显示的是过去一次的检测结果，程序只需要对旧的结果做访问控制。虽然有短暂拖尾，但实际效果不错，因为该线程运行时间通常比摄像头一帧时间短，结束后再换回新结果，几乎没有影响。还有两个 RectDetectItem 链表：recentDetectItems 和 thisDetectItems，分别用于记录最近十帧（可调）以内检测到的矩形、本次矩形检测的结果，矩形检测中间要用到之前链表数据，而不能包含本次检测结果，所以要放入另一链表，结束后再合并。

DoorMonitor 有许多控制检测效果的参数，分为几类：平滑、边缘检测及霍夫变换的各参数；矩形识别中用于合并残缺线的阈值参数；矩形识别中的宽高角度偏差、高边宽度长度阈值、能构成矩形的最大宽高长度偏差；判断门打开关闭的位移阈值；判断矩形连续性的长度比例、最少连续次数、最大超时帧数。

有几个布尔类型的设置参数，如：是否显示直线检测结果、标记检测结果、canny 边缘检测图像；是否进入测试模式；是否打开调试；是否保存结果到文件。

主要函数有：线程入口、霍夫直线边缘检测、矩形识别、标记识别、原始标记识别、门状态监测、监测范围选取、显示检测结果到窗口、窗口点击事件回调及命令行相关函数。原始标记识别在监测范围选取之后，在用户输入 dm range 命令后，将启动窗口点击事件在该模块的回调，用户可以在窗口内绘制矩形，在确认后，该矩形将作为监测范围，然后识别原始标记。这个过程结束后会重启程序。

显示检测结果到窗口函数根据设置参数，决定是否显示线段、监测范围、原标记位置、当前标记位置，用到的数据在 last_item 里。

2.4.2 线程流程

门监测线程在启动后就进入死循环，首先判断程序是否暂停和是否未清理数据，是则清理掉各检测项目，程序暂停时线程持续休眠。然后再判断不是空指针的条件下,执行：current_item = new LineDetectItem(camera->current, camera->frame_counter)。接下

来检测范围和原始标记是否为空，是则持续休眠。初始化做好就可以进入门状态监测了。门监测完毕后，因为本线程至此不再使用 current_item 所指的对象了，可以被其他线程访问了，所以将 last_item 销毁，重新指向 current_item 指向的对象。然后，在边框变化的室内行人检测方式中，还要进行室内行人检测，当边框图像突变时，修改行人检测标志，该标志在接下来的事件结束，或未发生事件时间超时时，修改回闲置状态，完成后，该线程又进入下一个循环，重复此流程。

2.4.3 初始化

本模块的初始化内容是：监测范围和原始标记位置，这些内容并不只是在线程启动时进行，而是只要用户执行“dm range”命令，就进行一次。步骤为：

（1） 窗口内绘制线框：主窗口点击事件注册的函数中调用了该模块的设置范围函数，当用户点击窗口并进行拖动时，可以在窗口内绘制出矩形；

（2） 确定监测范围：绘制出第一个矩形后命令行提示是否以此为监测范围，选 Y/y 确定，也可以反复绘制矩形，再点击确定；

（3） 识别原始标记：就是在 for 循环里不断地进行矩形检测，直到超时，期间检测到矩形时，用户输入确认后，以此为原始标记；

（4） 程序进程重启：因为初始化使得几个模块的原来的数据无效了，重启一次程序可以清除掉这些数据，注：重启并不是整个程序重新启动，而是执行了暂停、开始两个过程，在暂停时模块数据被清除了。

2.4.4 门状态监测

该部分的步骤如下：

（1） 边缘直线检测：采用高斯滤波、Canny 边缘检测、霍夫直线变换；

（2） 矩形识别：步骤：角度过滤、组合断裂线段、四条边和交点的位置判决；

（3） 标记识别：保存近期所有矩形信息、删除超时的位移和大小不连续项目。

（4） 门状态判别：门关闭时标记位移超过最小位移阈值，则判定门打开，门打开时标记必须在位置、大小上都非常接近原始状态，才判定为门已关闭。

（5） 上报门状态：向事件控制模块上报门是否打开，标记是否未检测到。

前三个步骤，由于内容很多，这里不细讲，详细请参见第三章。

门状态监测过程中依次记录了总次数、成功次数、成功前失败次数，用来计算几个指标：成功率、最大检测失败次数等，每一分钟过后都会重新记录。

2.4.5 室内行人检测

本来该部分内容应该独立于门监测模块的，但是其内容较少，并且需要在一个线程中执行，就放入的门监测模块的部分了。该部分的实现有两种方法：

（1）遮挡标记，标记识别失败次数过多，判定标记消失，事件控制模块就将其看作室内遮挡，认定为有室内行人经过，后续的开门事件视为出门。这一方法实现起来简单，但存在巨大漏洞：

a) 因为算法漏洞、环境变化或其他可能的原因，导致标记识别失败，程序将其错误判断为物体遮挡；

b) 门被打开至阈值之前，标记识别率降低，容易导致标记识别失败，错误判断为物体遮挡；

c) 入侵者可以从门缝处遮挡标记，导致识别失败。

如果上述三种情况伴随着门被打开，系统会将默认将其判断成出门，如果此时是入侵事件，将导致严重的后果。

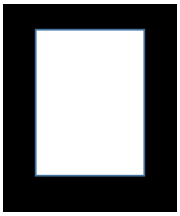


图 2.4.2 边框范围（黑色部分）

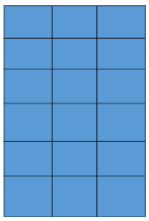


图 2.4.3 边框图像分割

（2）在除去光照的影响下，判断图像在外框范围内是否发生剧烈变化。外框是指一定边距以外的“回”字型范围，如图 2.4.2，假设整个矩形就是监控图像，黑色部分就是边框范围，算法需要该部分始终处在室内，而门始终在白色的区域。

算法的原理是：室内行人出门前必定进入监控画面，该边框范围内的图像会发生剧烈变化，这两个事件是有强关联关系的。将边框图像分成 4 个区域，左右是重点部分（实际上上下部分不做考虑），范围被用 X*Y(X<Y)网络线分割成一些小块，计算每个小块的像素颜色均值。这些数据保存在 curren_item 的一个二维数组中。门状态监测过程中，程序依次累加本次与上次数据的各小块均值之绝对差，再除以 X*Y，得到的颜

色变化量ΔC_1，再计算这两次图像的整体颜色变化ΔC_2，当$\Delta C_1 - m * \Delta C_2 > T$ (m:环境影响率，T:阈值)时，就判断有行人经过。

2.5 事件类型判别

2.5.1 主要结构

在 event_ctrl.h 文件下定义了事件控制类 EventCtrl，以及它的静态对象 event_ctrl。EventCtrl 没有线程入口，这里的事件控制都是在 DoorMonitor 上传门状态、室内行人遮挡状态的同时进行的。一次完整的，从门第一次打开，到记录视频结束的整个过程，就是一次事件，事件类型由宏定义为：NoEvent、EventLeaveHome、EventIn、EventGoHome、EventInvade、EventGoHomeContinue，当前类型保存在 status 变量中。门内事件类型分为：idle、shelt、leave,表示标记的遮挡情况，保存在 intra_status 中。

EventCtrl 下定义有许多超时时间参数，包括：detectFace_timeout（检测人脸超时时间）、safeEvent_timeout（出门或回家门打开超时，超过后会提醒用户忘记关门）、invade_timeout （发生一次入侵后，无论是否事件结束，在该时间范围内发生的所有出门事件都视作入侵事件）、invade_once_timeout（防止在入侵后录制时间过长）、leave_timeout（遮挡再离开后，超过 ~ms 状态置为闲置）、shelt_timeout（遮挡时间超过 ~ms，认为标记消失时间过长）、event_end_record_time（普通事件结束后需继续录制的时间）、exception_end_record_time（异常事件结束后需继续录制的时间）。

然后是一些标志变量和时间点记录：event_begin（事件开始标志）、event_end（事件结束标志）、event_is_multi（事件包含多次开门）、event_is_timeout（门打开超时）、time_event_begin（第一次开门的时间）、time_lastEvent_begin（上一次开门的时间）和 time_event_end（上一次关门的时间）、door_opened（门是否被打开）。

EventCtrl 下定义的大量事件处理接口，在门打开、关闭或超时时调用，功能通常是打印时间，对用户发起提示或警告等操作，如果软件后续联网，设计成远程交互，就可以从这里扩展。本程序设计有简易版本的远程交互，在以上事件发生调用接口时，向 QQ 中“我的 Android 手机”设备发送短消息，运行效果很好。

2.5.2 事件周期

当 DoorMonitor 检测到门打开时，将调用 EventCtrl:: doorOpening(), 当门首次打开（status == NoEvent）时，判断是否有室内行人遮挡（判断 intra_event != idle），是则设

置 status=EventLeaveHome，否则设为 EventIn，这时人脸识别线程将会开始任务，将 status 重新修改为 EventGoHome 或 EventInvade。然后，置 event_begin 标志为 true，event_end 标志为 false, time_event_begin 为当前时间，事件开始。

而如果 status == EventInvade 或 EventIn 时，status 不被修改，因为此类型为危险情况，否则事件进入复合状态（不止一次开关门）。status == EventGoHome 属于情况特殊，因为回家后遮挡的状态不能马上清除，故无法分辨出门、再次回家或是有入侵，将作为复合事件通知用户，始终放在 event 目录，但文件名会包含警示前缀。status == EventLeaveHome 时，由于人离开后遮挡状态立刻设置回空闲，可以直接进行下一轮判断。设置好 status 后，程序将 time_lastEvent_begin 更新为当前时间，再次将 event_end 标志设为 false，使得录制过程继续。

门打开时，如果门仍然打开，或标记消失，都视为门处在打开状态，此时计算自 time_lastEvent_begin 以来经过的时间，超时则会置 event_end=true，将 time_event_end 更新成当前时间。

当 DoorMonitor 检测到门关闭时，将调用 EventCtrl:: doorClosing(), 若之前门是打开状态，判断门打开是否打开超时，否，则同样置 event_end=true，将 time_event_end 更新成当前时间。若之前门是关闭状态，则更新室内标记遮挡情况，intra_event==shelt 时，判断出标记重新出现（遮挡物暂时离开），置 intra_event=leave；intra_event==leave 时并离开一段时间后，判断出标记识别恢复正常（遮挡物离开），置 intra_event=idle。

如果标记识别失败，将置 intra_event=shelt。

2.5.3 录制过程

录制过程在主线程中进行，当 event_begin 标志为 true 时，将其置回 false，然后主线程开始录制事件类型视频。如果 New.avi 没有录制完就发现录制时长已经达到，就只组合 Old.avi、New.avi 了，如果连 Old.avi 都还没有，就只是把 New.avi 的文件名作修改。反之，New.avi 录制完成后录制时长仍未达到，就新建文件：EventRest.avi, 直到事件结束且录制时长达到，开启新线程，将存在的临时文件组合成事件文件，保存在 event 或 exception 目录中。

值得注意的是，在判断事件结束且录制时长达到这个逻辑时，主线程与调用 EventCtrl 函数的门监测线程间会发生资源访问冲突，比如：主线程结束录像的同时，

EventCtrl 判断事件继续，要求主线程继续录像，于是我使用互斥锁控制访问，主线程判断录制是否结束前，对互斥锁上锁，如果同时门监测线程运行到事件开启程序段，将被阻塞，就保证了主线程的结束录像过程中资源不被修改，结束后置 event_ctrl.status =NoEvent，再释放锁，EventCtrl 不再判断事件继续，而是开启新事件。

2.6 人脸识别和对比

人脸识别采用的技术是 CascadeClassifier（滑动窗口+级联分类器），使用 haar-like 特征训练的分类器，本程序采用官方自带人脸识别分类器：haarcascade_frontalcatface.xml ，或着同类型的分类器，官方的开源分类器以及经过了很多次训练和使用检验，所以通常比自己训练的效果要好。

该模块有独立的线程入口，与 DoorMonitor 一样，进入循环后先判断程序是否暂停，是否要清理数据，人脸识别过程是工作在 event_ctrl.status ==EventIn 的条件下的，所以其他情况该线程会持续休眠。一次检测时长最多 8 秒（可设置），超时会直接将 status 设置成 EventInvade。为了防止反复开关门刷检测时长，本线程采用的是独立的计时器，在首次识别任务前置 is_detecting=true，记录当前时间，判断超时的时间起点就是这个时间，超时或识别任务完成后置 is_detecting=false，这是为了程序在下一次任务开始时能判断出识别开始标志，从而重新记录时间。

本模块同样采用了 last_item+current_item 的方法，不再赘述。

识别到人脸后，先按一定比例扩大选框，然后交由 Face++服务平台处理，Face++旷视科技有限公司旗下的新型视觉服务平台，提供了丰富的开源的人脸，手势等识别的 API 接口。调用人脸对比 API 的步骤如下：

（1）创建 face++账号（API Key），密码：API Secret；

（2）调用 Compare Face API，向服务器传输一个 JSON 文件，其中的数据包括 API Key、API Secret、两个待对比的图像（或一个图像与一个身份识别码）。

返回的 JSON 数据解析后就得到了一些信息，两者相似度超过阈值将被认定为同一身份。

授权操作需要依次使用命令：“face++”、“detect”、“promise”，有授权身份的人脸图像被保存在 face 目录，调用人脸对比 API 后，也会把授权人的身份识别码保存。

2.7 人机交互

系统采用命令行的交互方式，用户向命令行窗口输入命令，系统标准输出显示在窗口中。它的优点有：简单，容易开发；快速，在熟练使用后可以很快地进行各种操作；功能强大，比起图形界面交互方式，命令行模式提供了大量有用的命令和及其丰富的选项参数。缺点则是在很多时候操作麻烦，不适合大众。

本系统的命令行线程入口是 command 函数，它的程序实现简单粗暴，循环读取命令行里的一行字符串，用 if-else 语句一次判断输入命令的类型，这虽然很耗费时间，但对于交互过程来说，这点时间的影响微乎其微。

命令行系统使用 std::istringstream 工具输入数据，简洁方便。

```
a/save                -- 保存当前录像，开启新录像
q/quit                -- 保存当前录像，结束程序
start/stop/reload     -- 开始/暂停/重启程序
param                 -- 查看模块参数(不指定参数则默认全参数)
set                   -- 设置模块参数(带W的参数为可写,请在para
dm                    -- door_monitor  门状态监听模块
fd                    -- face_detector 人体目标识别模块
ec                    -- event_control 事件控制模块
cls                   -- 清屏
help                  -- 帮助，打开本菜单
```

图 2.7.1　命令列表

命令 set 和 param 的第一个参数是选择的模块，command 函数根据所选模块把任务委托给对应模块类的 cmd_set 和 show_param 函数。set/param 和模块名可交换位置。

```
[Home Moniter]dm param
Door monitor's params :
<1>是否进行高斯模糊                    [ true/false ]  (do_blur)                    true
<2>高斯模糊方差                        [ 建议0.8 - 2 ] (blur_sigma)                  1
<3>高斯算子大小                        [ 3或5,建议3 ]  (blur_ksize)                 5 x 5
<4>canny边缘检测阈值范围               [ 标准 20 80 ]  (canny_threashhold)          40 100
<5>霍夫直线检测阈值                    [ 标准 10 ]     (hough_threashhold)          10
<6>霍夫直线检测最短直线长度            [ 标准 2 ]      (hough_minlen_rate)          6
<7>霍夫直线检测允许缺口数              [ 标准 4 ]      (hough_mingap)               4
<8>标记高边最大两端点行间距            [ 标准 2 ]      (h_max_width)                3
<9>组合垂直直线时的最大缺口            [ 标准 0.2 ]    (h_unit_y_gap)               4
<10>原标记高边最短长度比例(基于范围高)  [ 标准 0.2 ]    (h_minLineLen_org_rate)      0.2
<11>宽边识别的最大角度偏差             [ 标准 40 ]     (w_angle_deviation)          25度
<12>宽边的最大长度偏差                 [ 标准 0.4 ]    (w_deviation_rate            0.32
<13>高边的最大长度偏差                 [ 标准 0.4 ]    (h_deviation_rate)           0.32
<14>比较原始位置与当前位置高边的最大长度偏差 [标准 0.4] (h_deviation_rate_cmp)       0.5
<15>门打开时判定再次关闭的最大位移（基于标记宽度和原始位置）[标准 0.05] (threashold_door_closed)       0.08
<16>门关闭时判定已打开的最小位移  （基于标记宽度和原始位置）[标准  0.2] (threashold_door_opened)       0.3
<17>判定标记消失所需检测失败次数          [ 标准 20 ]  (mark_disapeared_maxtimes)          20
<18>标记位置初始化的超时时间             (detectMark_timeout)                          1000ms
<19>最小识别帧间隔(工作频率)              [标准 1]    (min_frame_interval)               1ms
<20>一帧最大水平位移和宽度变化比例(基于宽)  [标准 0.25]   (threashold_rect_series_x)      0.3
<21>一帧最大垂直位移和高度变化比例(基于高)  [标准 0.15]   (threashold_rect_series_y)      0.2
<22>最小连续帧数，高于此则视为标记         [标准 3]     (threashold_rect_seriesCount)    3
<23>识别连续矩形帧超时                    [标准 10]    (rect_detect_series_frame_out)   15
```

图 2.7.2　dm param 命令输出内容

dm、fd、ec 命令是模块独立命令，command 将其委托给对应模块的 do_cmd 函数。

第 3 章　标记检测过程

3.1 高斯平滑滤波原理

高斯滤波目的是降噪，将图像模糊化，去除图像高频部分，因为一些细节度高的部分通常不是我们要检测的目标，容易被识别为伪边缘。其原理就是对图像进行加权平均，对于每一个像素点，都由其本身和邻域内的其他像素值经过加权平均，得到输出图像在该点的像素值。滤波器公式为：

$$G(x,y)=f(x,y)*H(x,y) \quad (1\text{-}1) \qquad H(x,y)=\frac{1}{2\pi\sigma^2}e^{\frac{(x^2+y^2)}{2\sigma^2}} \quad (1\text{-}2)$$

3.2 Canny 边缘检测原理

Canny 边缘检测算子是由 John F. Canny 在 1986 年提出的[1]，是常用的边缘检测算子，具有许多明显的优点，常被其他算子引为标准算子进行优劣对比，如：普通一阶差分、Sobel 算子、Robert 算子（交叉差分）等。Canny 算子符合 John F. Canny 提出的优秀边缘检测算子评价的 3 个标准[1]。

Canny 算法分为以下几步：

（1）计算梯度幅值和方向

假设图像平滑和二值化后的像素灰度函数为f(x,y)，其一阶偏导分别为：

$$G_x(x,y)=[\,f(x+1,y)-f(x,y)+f(x+1,y+1)-f(x,y+1\,]/2 \quad (2\text{-}1)$$

$$G_y(x,y)=[\,f(x,y+1)-f(x,y)+f(x+1,y+1)-f(x+1,y\,]/2 \quad (2\text{-}2)$$

梯度模：$$M(x,y)=\sqrt{G_x(x,y)^2+G_y(x,y)^2} \quad (2\text{-}3)$$

梯度方向：$$\theta[x,y]=\arctan\left(G_x(x,y)/G_y(x,y)\right) \quad (2\text{-}4)$$

（2）非最大梯度抑制

由步骤 2 得到的梯度二值图中的边缘通常不止有一个像素宽，而是很多个，所以需要对其进行边缘细化工作。为了达到标准 3 的要求，Canny 算法把局部中的非最大梯度抑制了，使得图像在边缘处只被检测一次，显得尖锐，结果清楚。

（3）双阀值

除梯度模低于低阈值的弱边缘被排除外，高于高阈值的强边缘也被排除，这有利

于缩小检测范围，在实际应用中，要对被检测目标的 边缘强度有先验知识，阈值取得太大或太小都可能导致成像不理想，边缘太多或目标边缘缺失。

（a）一阶差分　　（b）Sobel 算子　　（c）Canny 算子

图 3.1 三种边缘检测算子梯度二值图

对图 3.1 的三张二值化图进行比较，可以看出 Canny 算子是较为优秀的。

本程序使用的是 OpenCV 里的 Canny 函数，关键参数是控制效果的双阈值。

3.3 霍夫直线变换原理

霍夫变换是识别领域中对二值图像进行直线检测的有效方法，下面是霍夫直线变换的理论原理。

直角坐标系下直线的斜截表达式是：　　$y = \mathrm{kx} + \mathrm{b}$　　(3-1)

因为该方程无法表达垂直于 x 轴的直线，故需要更换参数表达式。

如图 3.2，将直线离原点距离定义为r，垂直线斜率定义为θ，

直线参数 k、b 与θ、r可互相转化：$\begin{cases} k = -\dfrac{\cot\theta}{\sin\theta} \\ b = \dfrac{r}{\sin\theta} \end{cases}$　　(3-2)

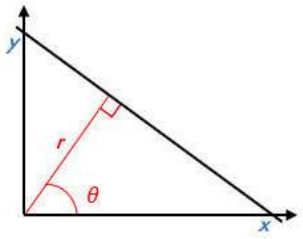


图 3.2 直线的r、θ定义

于是，直线表达式也可写成以r、θ为参数的形式：

$$\sin\theta * \mathrm{y} = (-\cos\theta)x + \mathrm{r} \qquad (3\text{-}3)$$

对(3-3)稍作变换，得：　　$r = x\cos\theta + \mathrm{y}\sin\theta$　　(3-4)

对给定的一点(x_0, y_0),可以将通过这一点的所有直线定义为：

$$r = x_0\cos\theta + y_0\sin\theta \qquad (3\text{-}5)$$

该表达式图像是一条正弦曲线，曲线上的某一点(θ', r')对应了直线：

$$\sin\theta' * \mathrm{y} = (-\cos\theta')x + r' \qquad (3\text{-}6)$$

该直线必定经过点(x_0, y_0)

通过上述步骤，我们完成了点-直线的对偶性变换，把$x \perp y$坐标系空间称为原空间，$\theta \perp r$坐标系空间称之为霍夫空间，原空间里的点(x_0, y_0)对应了霍夫空间里(3-5)表示的正弦曲线，原空间里的(3-3)表示的直线对应了霍夫空间里的点(θ, r) 。于是可以使用“投票机制”，遍历和计算霍夫空间里所有点或区域的票数，即经过的曲线数，得票数多的点或区域，再转换回原空间里的直线，就算作识别成功了。

对于给定点$x_0 = 8$ 和 $y_0 = 6$，可以在霍夫空间绘制出如图 3.3 所示平面图。

我们可以对图像中的所有点进行上述操作，如果由两个不同的点转换的正弦曲线在霍夫平面内交于同一点，这个点对应了原平面里的一条直线，这意味着原平面里的两个点经过了这一直线。例如，再对(12,3)、（9,4）转换并绘制得到图 3.4。

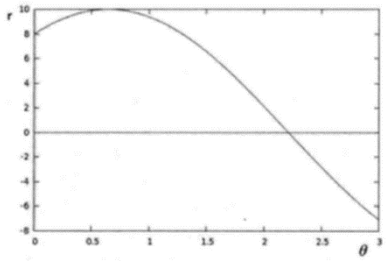


图 3.3 点(8,6)的霍夫空间图像

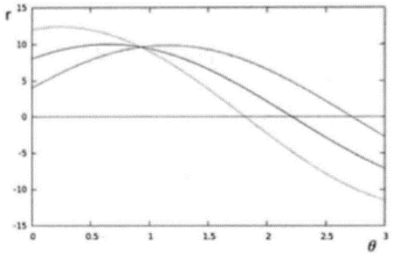


图 3.4 共线三点的霍夫空间图像

霍夫变换的编程实现思路就是：投票和搜索，建立表示霍夫空间的二维数组，统计每个点或区域的经过曲线数，较大者将转换回检测到的直线结果。

在实际的算法中，还要考虑离散化误差，单元格细分程度和交点个数阈值将决定统计精度，阈值不能设太低，否则导致误检，不能设太高，否则导致漏检。

本程序用到的是 HoughLinesP 函数，输入图像是 canny 边缘二值图，输出的线段数组保存在 current_item，可变参数有阈值、最小长度、最大缺失点数，

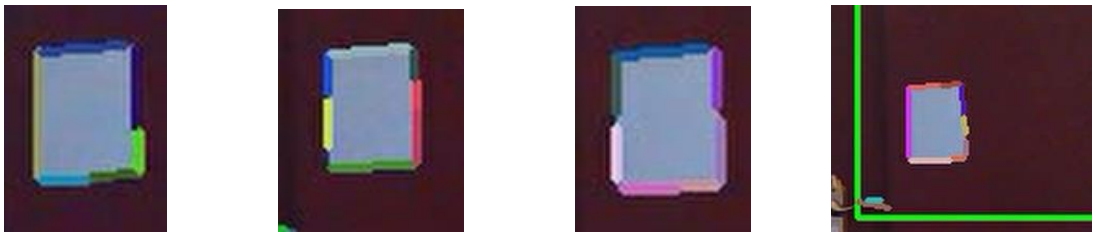
3.4 矩形识别算法

霍夫变换输出的线段数据是 Vec4i 类型，端点是(line[0],line[1])和(line[2],line[3])，识别矩形前，要对这些线段做处理。程序定义了辅助类：LineVData 和 LineHData, LineVData 以垂直线段（允许少量误差）构造，关键成员为上下点 y 坐标、高度。LineHData 以水平角度线段（允许角度偏差）构造，关键成员是左右端点和宽度。由于门打开后，垂直线段角度变化小，其允许角度偏差小，被看作是完全垂直的线段，即

不考虑宽度；而水平线段角度变化较大，其允许角度偏差通常大于 30 度，算法不会将其看作完全水平线，而是要带入左右端点计算。

函数输入为划定好范围后的图像、检测成功时的回调；输出为检测到的线段数组和一个被回调函数认可的矩形。

算法调用霍夫直线边缘检测后，首先过滤出水平和垂直线段，输出容器为链表，再分别对它们进行去重，即合并疑似同一边缘的残缺线段。



（a）　（b）　（c）　（d）

图 3.5 检测矩形前出现的残缺线段

算法 1：垂直残缺线段合并

```
输入：垂直线段链表
输出：list<LineVData> lineV_list
1: for 垂直线段链表里的每条线段 do
2:   if 宽度>max_w  and 高度>min_h
3:     构造 LineVData 对象 line1,声明 v_up、v_down
4:     将 is_up_exist 和 is_down_exist 置为 false
5:     for 垂直线段链表里的其他每条线段 do
6:       构造 LineVData 对象 line2
7:       if  |line1.x - line2.x| < 横坐标阈值
8: //经实验，残缺线段通常不超过 3
9:         if  line2 在 line1 上方且缺口小于阈值
10:          v_up= line2
11:          is_up_exist= true
12:        else if  line2 在 line1 下方且缺口小于阈值
13:          v_down= line2
14:          is_down_exist= true
15:        end if
16:        if is_up_exist
17:          line1.y_up = v_up.y_up
18:          line1.x = (v_up.x+ line1.x)/2
19:        end if
```

算法 2：半水平残缺线段合并

```
输入：水平线段链表
输出：list<LineHData> lineH_list
1: for 水平线段链表里的每条线段 do
2:   构造 LineHData 对象 line1,声明 h_left、h_right
3:   将 is_left_exist 和 is_right_exist 置为 false
4:   for 水平线段链表里的其他每条线段 do
5:     if 两者角度相差大 continue
6:     构造 LineVData 对象 line2
7:     if  line2 右端点与 line1 左端点距离小于阈值
8:       h_left= line2
9:       is_left_exist= true
10:    else if  line2 左端点与 line1 右端点距离小于阈值
11:      v_right= line2
12:      is_right_exist= true
13:    end if
14:    if  is_left_exist && is_right_exist
15:      line1.left  =  h_left.left  //左端点
16:      line1.right  =  h_right.right  //右端点
17:    else if  is_left_exist
18:      line1.left = h_left.left
19:    else if  is_right_exist
```

```
20:      //down 、 up&&down 同理
21:      line1 加入 lineV_list
22:      end if
23:    end for
24:  end if
25: end for
```

```
20:      line1.right=h_right.right
21:    end if
22:    line1 加入 lineH_list
23:  end for
24: end for
25:
```

注：line1、line2、v_up、h_left 等均为 LineVData 或 LineHData 对象

然后，开始识别矩形，方法是对四条边依次进行比较,算法如下：

算法 3：矩形识别

```
输入：lineV_list、lineH_list、检测成功时回调函数
输出：回调函数认可的矩形 result
1: for lineV_list 的每一个垂直线段数据 as lineV do
2:   for lineH_list 的每一个水平线段数据 as lineH do
3:    if lineH 在 lineV 右侧且缺口小于阈值
4:      计算 lineH 和 lineV 交点 node1
5:      if  node1 在 lineV 上方且缺口小于阈值
6:      lineH2_exist、lineV2_exist 置为 false  //左上关系
7:      for lineV_list 的每一个垂直线段数据 as lineV2 do
8:        if  lineV==lineV2 continue
9:        if lineH 在 lineV2 左，node 在 lineV2 上,缺口小于阈值
10:         lineH2_exist=true //左上右关系
11:     end if ; end for
12:     for lineH_list 的每一个水平线段数据 as lineH2 do
13:       if  lineH==lineH2 continue
14:       if lineH2 在 lineV 右，node 在 lineV 下,缺口小于阈值
15:         lineH2_exist=true //左上下关系
16:     end if ; end for
17: //若只识别了 3 条边，第 4 条边就和第 3 条比较
18: //仍只有 3 条边依然构建（可设置）
19:     if lineV2_exist && ! lineH2_exist
20:     for lineH_list 的每一数据 as lineH2(!=lineH) do
21:     if lineH2 在 lineV2 左，node 在 lineV2 下,缺口小于阈值
22:       lineH2_exist=true
23:     end if ; end for
24:     if  ! lineH2_exist
25:       以左上右方式构建 rect;移除 lineV2; break
26:     end if
```

```
27:     if lineH2_exist && ! lineV2_exist
28:     for lineV_list 的每一数据 as lineV2(!=lineV) do
29:     if lineH2 在 lineV2 左，node 在 lineV2 下,缺口小于阈值
30:       lineV2_exist=true
31:     end if ; end for
32:     if  ! lineV2_exist
33:       以左上下方式构建 rect;移除 lineH2; break
34:     end if
35:     if  lineV2_exist && lineH2_exist
36:       以全边方式构建 rect;移除 lineH2、lineV2
37:     end if
38:   else if lineH 在 lineV 右，node 在 lineV 下,缺口小于阈值
39:   //……左下关系，与左上关系同理
40:   else if lineH 在 lineV 左，node 在 lineV 上,缺口小于阈值
41:   //……  右上关系，同理
42:   else if lineH 在 lineV 左，node 在 lineV 下,缺口小于阈值
43:   //……  右下关系，同理
44:   end if
45:   if  rect 不为空
46:     if 回调() == true
47:       result=rect; return
48:     end if
49:     移除 lineH、lineV; break
50:   end if
51:   end for
52: end for
```

注：lineV、lineV2、lineH、lineH2 均为 LineVData 或 LineHData 链表的遍历器

原始标记识别和运行中标记识别的回调函数不一样，前者在识别到矩形后会等待用户确认：是否将该矩形作为原始标记位置？后者则要先过滤与原始大小差别过大的结果，过滤颜色差别过大的结果（可选），然后开始标记识别算法。

3.5 标记识别算法

标记识别的原理是利用短暂时间内标记不可能位移太远，以及干扰检测的矩形位置和大小随机的特性。

在单个矩形识别成功时调用回调函数，将该矩形与所有历史矩形数据比较，若发现连续则更新那个历史数据，其矩形改为该矩形，帧数改为当前帧，连续性计数 seriesCount 加 1。若该矩形没有与任何历史矩形连续，则将其加入链表 currentDetectItems。

在矩形识别算法结束后，再将 recentDetectItems 中的未更新的帧数超过阈值（超时）的数据删除，若已经有数据的连续次数达标（seriesCount），就把它记为标记，并删除其他所有历史数据（这有利于提高识别率，减少误判率），否则将新的数据加入。

第 4 章　系统性能测试

4.1 应用程序生成

项目文件列表详情见附录 1 图 5.6。

图 4.1.1 项目调试

图 4.1.2 项目程序生成

在项目开发中的进行测试，通常不生成软件，而是在 IDE 内进行调试，如图 4.1.1，点击开始调试（S）。

开发结束可以发布应用程序时，在菜单栏里选择生成 -> 生成 HomeMonitor，已经发布程序需要再次编译和发布时，就在菜单栏里选择生成 -> 重新生成 HomeMonitor，生成的应用程序目录见图 4.1.3。

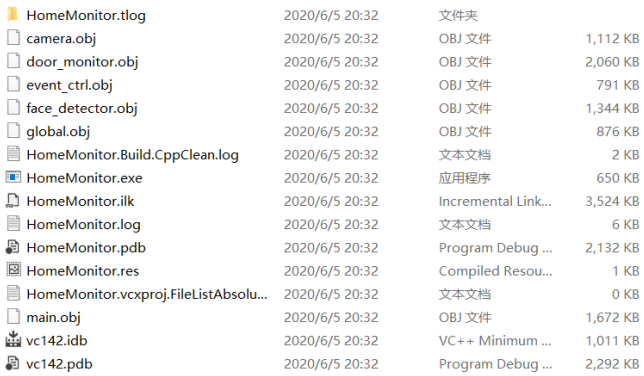


图 4.1.3 项目生成文件列表

其中的 HomeMonitor.exe 就是可执行的应用程序，其需要的运行环境为 Windows X64，暂时没有提供 X86 的版本，因为我的 OpenCV 版本是 X64 的，后续可以下载 X86 的版本，然后就可以生成 X86 版本的应用程序了。

注意：在运行程序之前，应确保摄像头没有被其他程序占用。

4.2 门监测模块调试

门监测模块的性能是决定设计最终效果好坏的关键因素，所以需要对门监测模块进行测试，步骤如下：

（1）确定监测范围、设置测试模式

在程序首次打开时，固定使用一次“dm range”命令，程序进行中可以随时更改监测范围，更改后程序会重启。命令下达后，用户在窗口中绘制监测范围矩形边框，如果不满意可以重新绘制，绘制完成后在命令行中输入 Y/y，或直接回车，就表示确认。监测范围就被设置成置成这个矩形了，如图 4.2.2（a）中的绿色矩形边框。然后开始识别标记初始位置，即图 4.2.2（a）中的浅蓝色矩形边框。dm show org 和 dm show range 命令可以用来在图像中显示原标记位置和监测范围。

输入“dm test”命令进入测试模式，阻止模块上报事件。

（2）门关闭时的测试

如图 4.2.2（c）所示，使用命令“dm show （mark）”后，标记的监测结果就是红色矩形框，偶尔会出现大小不对等的情况。输入命令“dm show canny”后，canny 边缘检测二值图就显示在了新窗口中，见图 4.2.3，如果 canny 图不理想，可以调整边缘检测参数，或者采用移动摄像头的办法。

输入命令“dm show lines”，查看霍夫变换的直线检测结果，如果觉得效果不理想，可以调节霍夫变换阈值、最小线段长度、最大缺口数三个参数，根据标记边缘线段的断裂情况、缺口情况，可以调节矩形检测中的各项偏差参数数值。测试中发现，当外界干扰很多时，霍夫变换结果中标记边缘的直线残缺程度加重了，如图 4.2.3（d），这一点暂时不清楚原因。

图 4.2.4 中的结果是模块指标，需要提前输入命令“dm debug”，前面的时间是指一次完整门监测程序用时，可以看到，正常情况下的用时是小于 33ms（1/fps）的，工作频率高，就不会遗漏某一帧画面。

经测验，如果外界干扰很少或没有，识别成功率很高，失败次数很低，在光照强度合理时，识别率几乎达到 100%，但如果阳光直射，或者光线不足时，识别率会下降

到 80%，甚至更低。如果外界干扰大量增加，识别率下降到 80%，这一问题说明矩形与标记识别的某些地方需要改进。因为干扰增多通常与室内经过的行人有关，这种情况下接下来的开门事件通常不会是入侵事件，所以其实对安全性影响不大。

（3）门打开时的测试

门已经被判定打开以后，后续的标记识别率对事件判定结果没有影响，因为事件控制模块会忽略门打开后的标记消失，当标记消失时，依然按门正打开处理，直至标记归位。我们的目标是测验标记位移至门被判定打开的时候的标记识别情况，此时的识别率对门打开事件的判断有重要影响。

门打开判定阈值比例默认设置为 0.3，将门打开到标记水平位移超过 1 倍标记宽的位置，图 4.2.5 显示的结果表示系统能够在门打开后识别到标记，且效果较好。

如图 4.2.6 所示，们打开后标记识别率显著降低，但这没有什么问题，因为系统只需要识别到一次标记位移的情况，就可以判定门打开了。经测验，当识别率高于 50%时，几乎不可能有漏检的情况。因为门打开前可能先出现标记消失的现象，可能是人为或亮度变化等原因，总之，shelt_is_leave 模式是存在巨大漏洞的，我已将其替换。

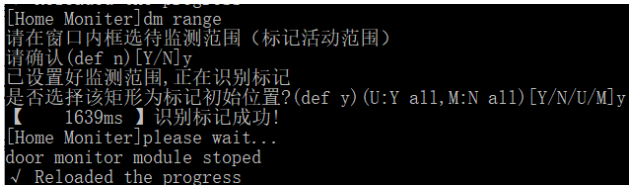


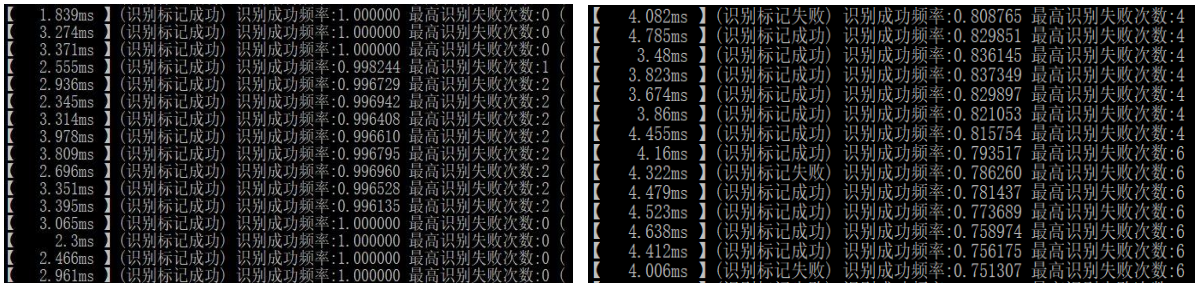
图 4.2.1 dm range 命令

(a)初始位置　　(b)原图像　　(c)检测结果

图 4.2.2　门关闭时的标记检测

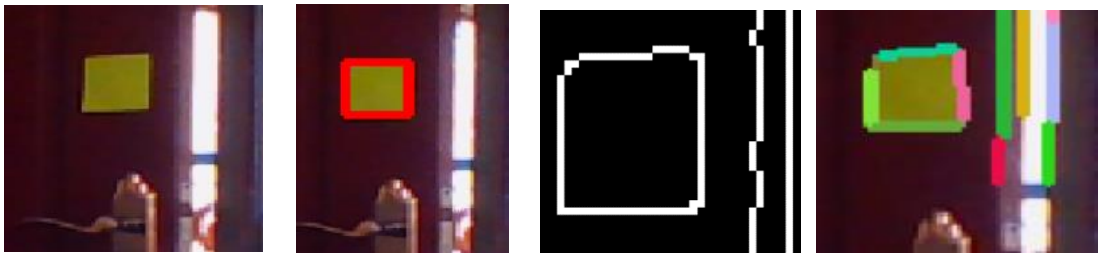
(a)canny 边缘　　(b)直线检测　　(c)存在干扰时的边缘　　(d)存在干扰时的直线检测

图 4.2.3 门关闭时的直线边缘检测



(a)无干扰情况　　(b) 有干扰情况

图 4.2.4 门关闭时调试结果



(a)原图像　(b)检测结果　(b) canny 边缘　(b) 直线检测

图 4.2.5 门打开时标记检测

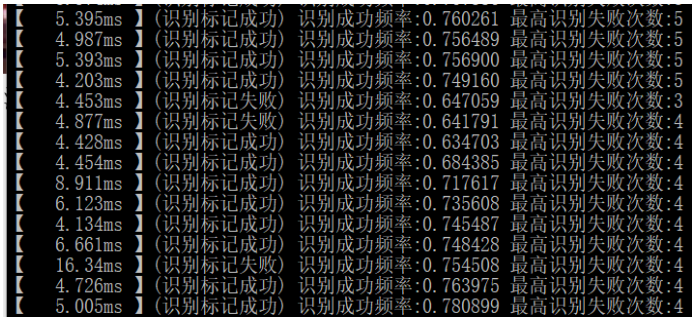


图 4.2.6 门打开时调试结果

门监测模块在不同环境都调试过，经反复验证，门状态监测是有效且基本稳定的，具体检测结果见表 4.1，每一次检测都用时 30 秒，都在无遮挡、无门移动的条件下进行，最大允许标记识别失败数为 20。注：门打开后识别失败是很正常的。但房屋出入口门 2 因为是在光线较强的中午进行的，识别率偏低

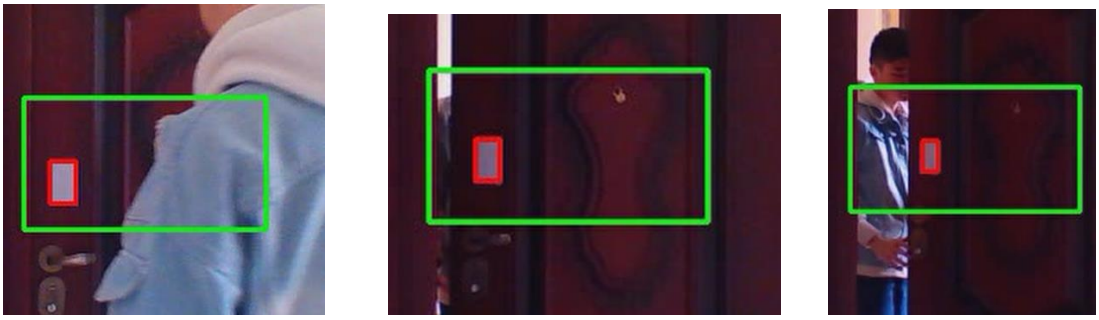
			门关闭时			门打开后		
测试对象	时间	标记类型、颜色	标记识别成功率	最高识别失败次数	是否出现标记识别失败	标记识别成功率	最高识别失败次数	是否出现标记识别失败
卧室房门 1	下午	白色纸片	98.1%	2	否	74.2%	7	否
卧室房门 2	下午	白色纸片	94.6%	2	否	63.9%	11	否
卧室房门 1	晚上开灯	绿色便签	100%	0	否	64.1%	13	否
卧室房门 1	下午开灯	黄色便签	96.0%	3	否	59.7%	14	是
房屋出入口门 1	早晨	白色纸片	99.1%	1	否	69.7%	7	否
房屋出入口门 1	早晨	绿色便签	94.1%	2	否	75.2%	6	否
房屋出入口门 2	中午	绿色便签	82.3%	5	否	45.2%	18	是
房屋出入口门 2	中午	红色便签	85.2%	5	否	51.3%	16	是

表 4.1 各种情况下的门状态检测结果

4.3 事件调试

为测试事件识别的效果，首先输入“dm untest”取消测试模式。门监测初始化完毕后，就可以直接进行测试了。测试内容有这几部分：人从室内离开、人从室内离开后立即进入室内、授权人从室外进入，未授权人从室外进入并随即离开。

图 4.3.1 中，当门打开角度偏大时，仍能识别到标记，入侵者的开门速度即使很快，一般情况下都能检测到门被打开，因为当门活动到图（b）的位置时，标记的横坐标位移就大概超过阈值了，而且因为门刚打开时的角度很小，横坐标方向的加速度分量和速度分量是比较小的，这为标记识别提供了良好的环境。



（a）门关闭时　　（b）门刚打开　　（c）门已打开

图 4.3.1　门打开过程中的识别到的标记

以下是各种事件的测试结果：

（1）入侵事件，当入侵者进入房间后，门再次被打开了，假设按正常的事件判断，第二个事件应该是出门，但这很明显是危险事件，程序将这些开门全记为入侵事件期间发生的。入侵已经发生，后续是什么类型已经不重要了。应当立刻报警。

通常情况下，这第二次开门事件表示入侵者已经开门离去。

```
[Home Moniter]2020-5-25【9-34-13】Warring:  门被打开
2020-5-25【9-34-21】@@@ 注意：入侵事件发生
2020-5-25【9-34-33】  入侵事件:门已关闭,用时：20.27s
2020-5-25【9-34-36】有室内物体遮挡
2020-5-25【9-34-38】遮挡物暂时离开
2020-5-25【9-34-47】重新遮挡
2020-5-25【9-34-48】遮挡物暂时离开
2020-5-25【9-34-52】  入侵事件:门再次被打开
2020-5-25【9-35-10】  入侵事件:门已关闭,用时：17.89s
2020-5-25【9-35-35】有室内物体遮挡
2020-5-25【9-35-36】遮挡物暂时离开
2020-5-25【9-35-51】遮挡物已离开
```

图 4.3.2.1 入侵事件的命令行结果

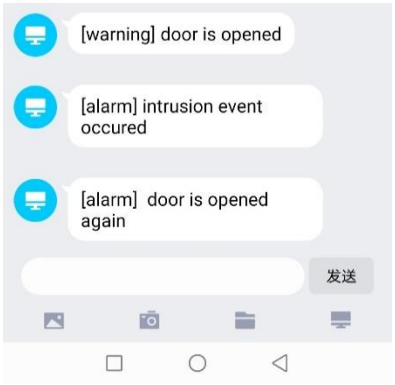


图 4.3.2.2 入侵事件手机接收到的消息

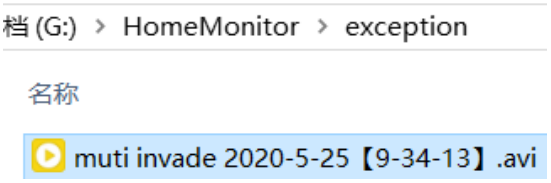


图 4.3.2.3 入侵事件保存的视频文件

（2）回家事件，即使识别到人脸且身份识别成功，依然会向 QQ 发送警告门被打开的消息。此时如果主人在外，明确此时家人不应该回家，而是人脸识别对比阶段出错（可能是入侵者的手段），就可以联系保安，或者报警了。

```
[Home Moniter]2020-5-25【9-31-29】Warring:  门被打开
[ INFO:3] Initialize OpenCL runtime...
2020-5-25【9-31-31】欢迎回家
2020-5-25【9-31-35】  回家事件:门已关闭,用时：6.514s
2020-5-25【9-31-37】有室内物体遮挡
2020-5-25【9-31-38】遮挡物暂时离开
New Event file: G:\HomeMonitor\event\gohome 2020-5-25【9-31-29】.avi
[Home Moniter]
```

图 4.3.3.1　回家事件的命令行结果

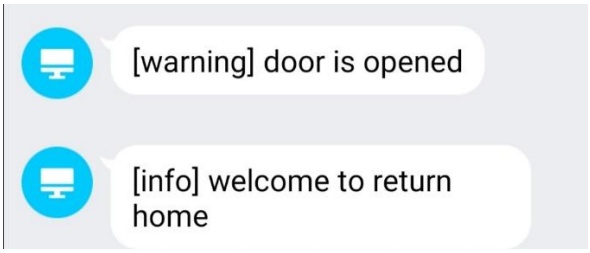


图 4.3.3.2 回家事件手机接收到的消息

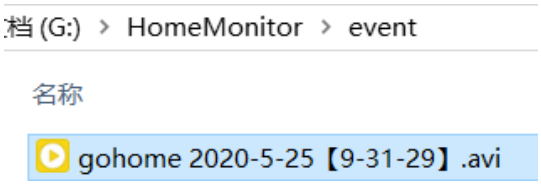


图 4.3.3.3 回家事件保存的视频文件

（3）出门事件，且忘记关门。门打开时间太长，就会提示用户忘记关门。

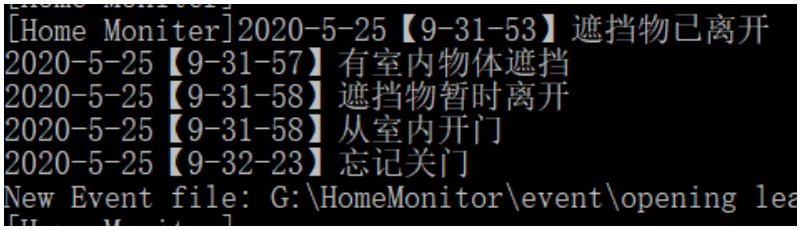


图 4.3.4.1 出门忘记关门的命令行结果

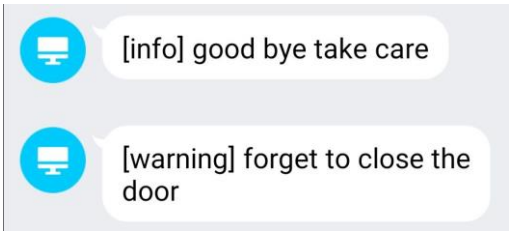


图 4.3.4.2 出门事件手机接收到的消息

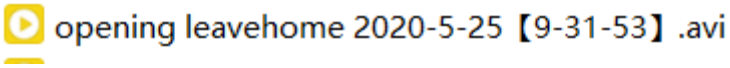


图 4.3.4.3 出门事件保存的视频文件

（4）出门后再次回家的复合事件，leaveHome 的发生不影响后续的事件判断，保存视频时的事件标志是以最后一次为标准。

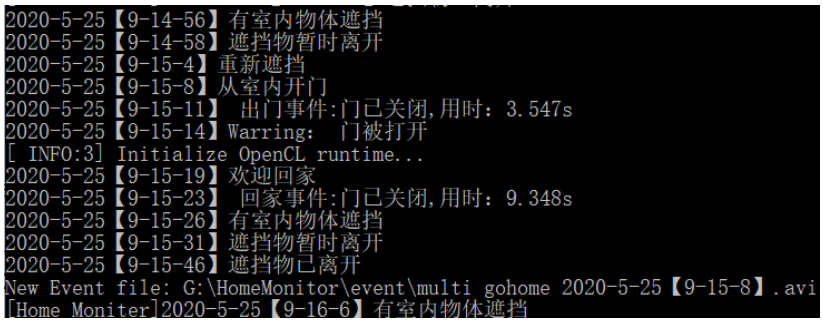


图 4.3.5.1　leaveHome、goHome 复合事件的命令行结果

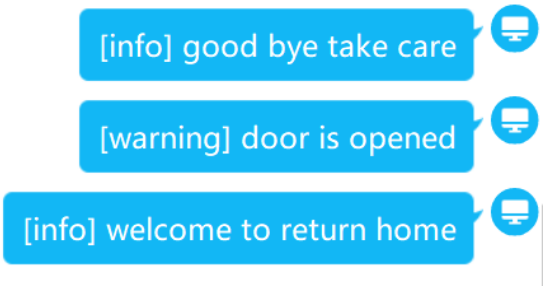


图 4.3.5.2 复合事件电脑向手机发送的消息

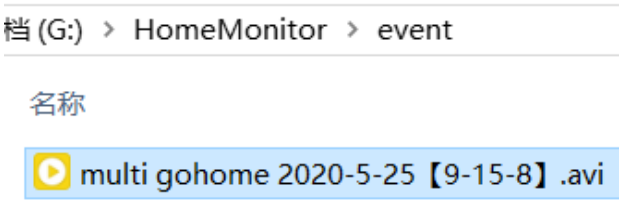


图 4.3.5.3 复合事件保存的视频文件

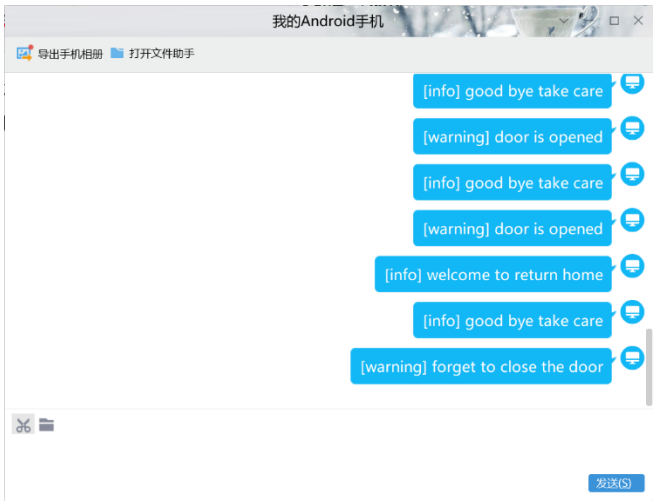


图 4.3.4 电脑端向手机发送事件消息

如图 4.3.5，可以看到，当事件发生时，电脑向手机发送的消息会出现在通知栏里，这样，用户可以在第一时间收到当前房屋的安全情况。

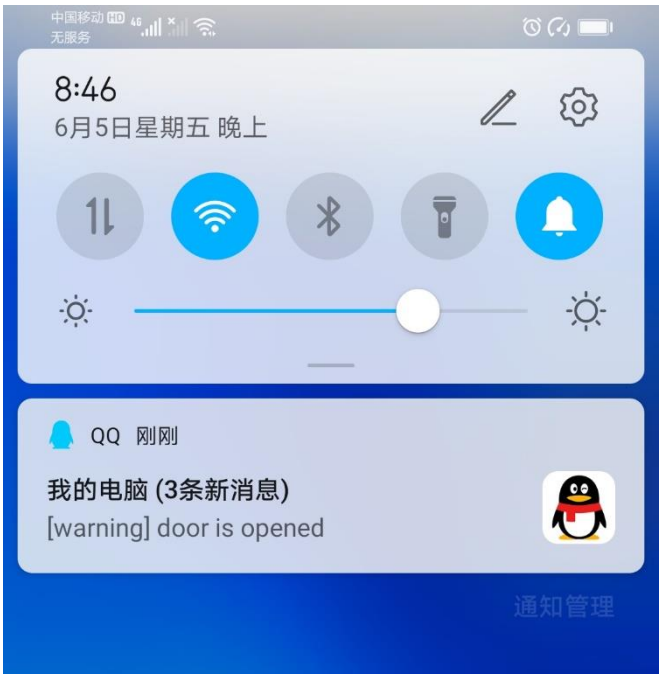


图 4.3.5 手机接收的事件消息出现在通知栏

事件测试在不同环境都进行过，经反复验证，稳定性很高，可以适应。具体检测结果见表 4.1。测试同一使用绿色标签作为标记。

测试环境	时间	事件 1	事件 1 情况	事件 2	事件 2 情况	事件 3	事件 3 情况
卧室房门 1	中午	出门	5 次全成功	入侵	5 次全成功	回家	5 次全成功
卧室房门 1	下午	出门忘记关门	5 次全成功	入侵多次	成功	回家复合	成功
卧室房门 2	中午	出门忘记关门	4 次全成功	入侵	有一次判定成了出门	回家	有一次识别身份失败
卧室房门 2	下午	出门	全成功	入侵多次	全成功	出门回家	成功
房屋出入口防盗门 1	中午	出门	全成功	入侵超时	全成功	回家忘记关门	手机一次没接收到消息
房屋出入口防盗门 1	晚上	出门忘记关门	全成功	入侵	全成功	回家	全成功
房屋出入口防盗门 2	中午	出门	手机一次没接收到消息	入侵多次	有一次判定成了出门	回家忘记关门	全成功
房屋出入口防盗门 2	晚上	出门	全成功	入侵超时	全成功	回家复合	全成功

表 4.2 各种情况下的事件检测结果

经反复实验，容易出现的错误识别情况主要有 2 种：一是入侵事件识别成出门，这是由于采用了标记消失判定室内行人遮挡的模式，如果标记识别成功率较低，就很容易出现这种情况。二是识别身份失败，导致回家事件判定成了入侵，这很可能是人的行动速度过快，或角度不适合，导致识别人脸失败，或是识别的人脸不理想，人脸对比的相似度结果较低。

因为向手机发送 QQ 消息需要用户先将窗口打开并点击对话框确定，出现操作失误则会导致发送消息失败。在调试中，发现也偶尔会出现这样的情况：某次消息已经被写在对话框里了，但模拟键盘的回车键并没有把消息发送出去，而是换行，直到下一次消息准备发送时，这两条消息才被一起发送出去，这就造成了接收消息不及时。

结论

无论做什么，安全总是第一位。我设计的家庭入侵检测系统，是一款安全防护软件，能在入侵发生时，快速识别到出入口房门被打开，在识别到陌生对象人脸或长时间未识别到人脸后，向用户发出入侵警告，把入侵的全过程录制下来并保存。除此之外，为了不打扰房屋主人正常的出入活动，系统的功能就不仅仅是入侵检测，而要分辨出不同的事件，人脸识别对比把身份区分开了，主人可以正常进入房屋；室内行人经过时，系统能够捕捉到这个动作，从而将下一次开门事件记为“出门”，如果匆忙离开忘记关门，系统也能提示用户。这些功能经过了反复测验，正常情况下，系统都工作得很好，误报、漏报很少见。

优秀的指标得益于门监测算法的巧妙设计，Canny+Hough 的经典边缘检测算法清晰地勾勒出图像里的直线轮廓，在门监测模块中，我用了大量的篇幅编写矩形识别、标记识别算法，首先对霍夫变换的直线数组结果进行了处理，过滤角度，组合残缺线段，这些预处理是矩形识别成功率高的前提。对于一个矩形的 4 条边，两两组合有 4 种方式，每一种方式下的两边再跟另外两边组合比较，三边组合就有 8 种方式，只组合三边不够精确，算法还尝试对两个新边进行组合，从而使四边都适配成功。

矩形识别的结果不一定是“标记”，所以算法还利用标记的位移连续性，将其与其他干扰矩形区分开，诚然，算法并不能区分标记和图像中存在的矩形物体，但程序其实不需要在门打开后区分它们，只要那个矩形物体不与原标记位置重合，其他位置识别到的矩形并不会使系统误判出关门事件。重要的是，用户选择监测范围是必须注意：范围内（通常在门扇上）不得出现第二个矩形物体，否则在门关闭时容易误判开门事件，基于这一点，我将对原标记识别算法做改进：检测范围内所有矩形，一旦数量不为 1，不允许门监测线程运行。

程序存在一个致命的 BUG，入侵者可以利用门监测的识别原理：标记的位移超过阈值时，才视为门已打开。换句话说，只要标记在原位置不移动，那么系统都不会检测到门打开，基于次，入侵者可以先打开一个门缝，刚好使手穿过去，先用反光镜（或手机）观察标记大小、位置，然后制作一个相同的标记，或者在不遮挡条件下直接把标记从门上扯下来。入侵者用两根手指捻起这张矩形纸条，在保持其不移动的同时，打开房门，进去后再贴回去。在这整个过程中，标记在图像中的位置始终没变，

但门已经被打开，入侵已经发生。对于这样的程序漏洞，我的解决办法就是：对门监测系统升级，放置多个标记，而不是只有一个，检测系统要检测多个标记，任意一个标记发生移动，就视为门已打开，入侵者同时对这么多标记下手是极其困难的。

实际上，图像处理形式的门监测系统，把问题还是复杂化了，总有可能被钻漏洞，不能确保万无一失。如果采用物理方案，问题就简便得多，比如在门上安装某种机关装置，门打开后就直接触发，这样的方案既高效，又稳定。但在我看来，研究科学技术不该太过急功近利，这些新科技可能暂时在实用性上比不过传统技术，但这只是因为不够成熟，或者应用的位置不恰当，待新科技有了充分发展，其潜能是无穷的。

参考文献

[1] Canny J．“A computational approach to edge detection”．Pattern Analysis and Machine Intelligence，IEEE Transactions on，1986 (6)：679—698．

[2] 刘智.一种基于 Hough 变换的四边形分类识别算法[J].广西科学院学报,2010,26(04):412-414.

[3] Rafael C.Gonzalez、Richard E.Woods，《数字图像处理（第三版）》 [M].电子工业出版社，2017：443~492

[4] 白松让,段敏,曹景胜,郑苏,杨晓丽.基于 OpenCV 的车道线智能检测和识别[J].辽宁工业大学学报(自然科学版),2020,40(02):92-95.

[5] 王丽.基于霍夫变换的证件图像尺寸恢复[J].福建电脑,2014,30(04):116-119.

[6] 张清华,林勇,温阳东.基于 OpenCV 的家庭入侵检测系统[J].制造业自动化,2018,40(03):78-80+113.

[7] 郭逸伦.基于 OpenCV 的边缘检测算法效率分析[J].科学技术创新,2019(01):87-88.

[8] Adrian Kaehler、Gary Bradski，《OpenCV3（中文版）》[M].清华大学出版社，2018

[9] CSDN 博主「YuYunTan」．经典霍夫变换（Hough Transform）[EB/OL] .https://blog.csdn.net/yuyuntan/article/details/80141392, 2018-04-29

[10] CSDN 博主「leonardohaig」．霍夫变换直线检测（Line Detection）原理及示例[EB/OL] .https://blog.csdn.net/leonardohaig/article/details/87907462, 2019-02-24

[11] 朱娟,刘艳滢,王延杰.一种基于 Hough 变换的新直线段检测算法[J].微电子学与计算机,2008,25(12):60-63.

[12] 张国福,王呈.基于改进概率霍夫直线检测的电梯门状态检测方法[J].南京理工大学学报,2020,44(02):162-170.

[13] 马凌云. 机器学习在智能家居监控系统中的应用[D].北京交通大学,2019.

[14] 马艳,张治辉.几种边缘检测算子的比较[J].工矿自动化,2004(01):54-56.

[15] 林卉,赵长胜,舒宁.基于 Canny 算子的边缘检测及评价[J].黑龙江工程学院学报,2003(02):3-6+16.

[16] 王耀贵.图像高斯平滑滤波分析[J].计算机与信息技术,2008(08):79-81+90.

[17] 郑清超. 基于红外图像分析的入侵探测系统研究[D].华南理工大学,2011.

[18]路文超,权伟,杨雯,郭培峰,和腾龙,杨应山.一种基于红外探测技术的住房入侵检测系统[J].软件工程,2017,20(03):42-44+5.

[19]吴双. 基于 MFC+OpenCV 的视频监控区域入侵检测系统设计与实现[D].山东师范大学,2017.

[20] 张少伟. 基于机器视觉的边缘检测算法研究与应用[D].上海交通大学,2013.

[21] 孙丰荣,刘积仁.快速霍夫变换算法[J].计算机学报,2001(10):1102-1109.

[22] 张勇红.基于霍夫变换的铭牌 OCR 图像旋转矫正方法[J].电测与仪表,2015,52(08):125-128.

[23] Spyretta Golemati,John Stoitsis,Emmanouil G. Sifakis,Thomas Balkizas,Konstantina S. Nikita. Using the Hough Transform to Segment Ultrasound Images of Longitudinal and Transverse Sections of the Carotid Artery[J]. Ultrasound in Medicine & Biology,2007,33(12).

[24] A.L. Kesidis,N. Papamarkos. A window-based inverse Hough transform[J]. Pattern Recognition,2000,33(6).

附录（项目文件展示）：

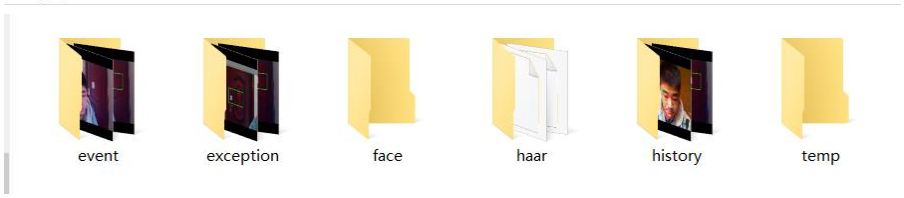


图 5.1　根目录

图 5.2　事件记录

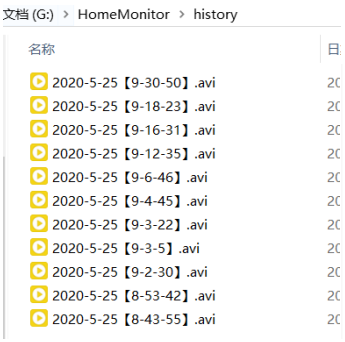


图 5.3　历史文件

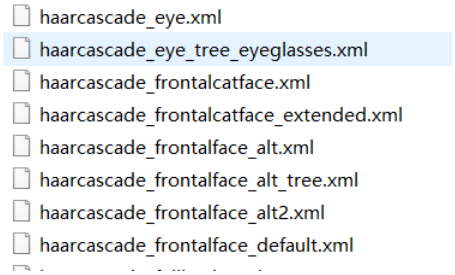


图 5.4　haar 分类器

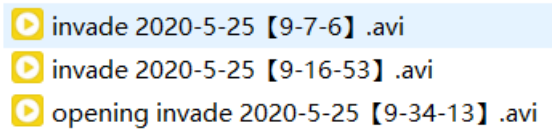


图 5.5　异常（入侵）记录

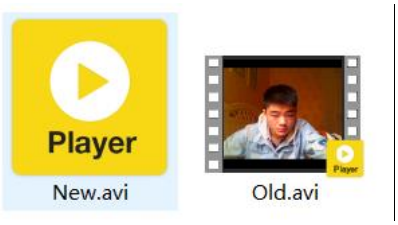


图 5.6　临时文件

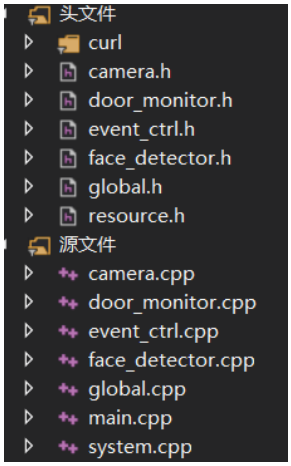


图 5.7　项目结构