

文章编号:1005-0523(2012)03-0021-05

基于 OpenCV 和 DirectShow 的交通运动目标检测算法研究

刘馨¹, 马宏锋^{1,2}, 王宏斌¹

(1. 兰州工业学院电子信息工程系, 甘肃 兰州 730050; 2. 兰州交通大学电子与信息工程学院, 甘肃 兰州 730070)

摘要:针对运动目标的自动检测问题,提出了一种改进的背景差分算法,采用背景图像多帧平均进行背景的获取和更新,进一步检测出运动车辆轮廓。使用 VC++6.0 作为开发环境,结合 DirectShow 视频处理库和 OpenCV 图像处理库进行编程实现。通过对两个不同场景的实验结果表明该算法能正确地检测出运动车辆,能够满足实时性的要求。

关键词:OpenCV; DirectShow; 背景差分法; 背景多帧平均

中图分类号:TP37

文献标志码:A

随着我国经济建设的快速发展,城市人口和机动车辆急剧增加,交通拥挤堵塞严重,因此如何有效管理交通信息对解决城市交通问题起到越来越重要的意义。由于运动目标的视频图像序列中提供了丰富的信息,这是智能交通监控系统研究的热点,其中如何准确地检测出视频图像中的运动目标直接影响监控系统的效果。

本文采用 VC++6.0、DirectShow 和 OpenCV2.0 技术,实现了一个基于视频的交通车辆检测系统,主要工作体现在以下几个方面:①采用 DirectShow 技术获取视频信号中的图像帧;②采用 OpenCV 库对图像进行处理并提取出感兴趣的区域;③采用基于多帧平均算法的背景差分法将感兴趣区域中的运动车辆检测出来。

1 开发环境介绍

1.1 DirectShow 介绍

DirectShow 是微软公司在 ActiveMovie 和 Video for Windows 的基础上推出的新一代基于 COM(component object model, COM)的流媒体处理的开发包,采用 Filter Graph 来管理整个数据流的处理过程,可从采集卡上捕捉数据并用于后期处理^[1]。DirectShow 可将应用程序和硬件设备隔离开来,它即支持各种媒体格式(如 Avi, Mp3, Asf 等),也支持 WDM 和 VFW 驱动的采集卡,为媒体流的采集提供有力的支持。

DirectShow 的基本构建模块称为滤波器(Filter)组件,DirectShow 可将数据流的处理过程分解成若干步骤,每一步由一个 Filter 来完成。Filter 可根据功能的不同分为 3 类:Source Filters, Transform Filters 和 Rendering Filters。Source filters 主要负责取得数据,数据源可以是文件、因特网、或者计算机里的采集卡、数字摄像机等;Transform Filters 主要负责数据的格式转换、数据流的分解或集合、传输;Rendering Filters 主要负责将数据送给声卡、显卡进行多媒体演示,或者将输出文件存储或发布。Filters 之间通过 Pin 进行连接构成滤波器图(Filter Graph),通过 Filter Graph Manager 控制一系列的数据处理过程^[2]。此时的媒体数据已经转换为一帧一帧的图像,更方便一张一张捕捉下来。

收稿日期:2012-04-06

基金项目:甘肃省自然科学基金项目(096RJZA084);甘肃省科技支撑项目(1104GKCA032);兰州市科技计划项目(2010-1-225)

作者简介:刘馨(1983—),女,助教,硕士,研究方向为通信与信息系统。

1.2 OpenCV 介绍

OpenCV 是 Intel 开源计算机视觉库,主要由一系列 C 函数和少量的 C++ 类构成,实现图像处理和计算机视觉方面的相关算法。OpenCV 独立于操作系统、硬件和图形管理器,具有通用的视频图像载入、保存和获取模块,便于实时应用程序的开发。

OpenCV 中封装了很多针对视频流的处理函数,主要有以下 3 种^[3]:

1) 利用 HighGUI 库实现对视频的输入、输出及处理,由于 HighGUI 库为用户提供简单的用户接口,方便获取摄像头和采集卡的视频流;

2) 由于 OpenCV 集成了 DirectShow 模块,可采用该技术进行视频流处理;

3) 采用 OpenCV 提供的 CvCam 库对视频流进行处理。

这 3 种方法相比,HighGUI 只是用来建立快速软件原型或试验用,在生产中应用还有待完善,CvCam 最初作为多摄像头同步使用,在今后的 OpenCV 版本中逐步淘汰,故本文采用 DirectShow 模块进行视频流采集和处理。

2 车辆检测模块的设计

2.1 车辆检测模块的算法流程

本文设计的车辆检测模块是由采集设备和基于 PC 的软件处理系统组成,完成在 PC 机上进行现场采集,并实时处理视频图像的目的。从功能上来说该模块主要有 3 部分构成:

1) 视频发送:主要负责提供视频图像信号;

2) 视频采集:视频采集卡用 PC 机上的软件封装;采集到的原始视频信号将暂时存在 PC 机的内存上,由负责视频处理的 Filter 在内存中直接读取并进行相关算法处理,这样可以绕过硬盘读取速度的瓶颈,降低对硬件的依赖;

3) 视频处理:将背景差分算法结合 OpenCV 库进行模块化封装。

2.2 图像采集模块的设计

图像采集模块是将摄像机传输过来的视频图像通过视频采集卡将其数字化,按每秒 1~25 帧的速率捕捉图像传输给 PC 机进行相关处理并显示。DirectShow 定义了标准组件来处理流媒体数据,开发人员只需要调用 API 函数来实现对流媒体的控制,处理一般的视频捕捉问题。

这里定义了 CDXGraph.cpp 类封装了 Filter Graph^[4],

```
CDXGraph * m_FilterGraph;           // Filter Graph 封装
m_FilterGraph->Creat();               // Filter 初始化方法
m_FilterGraph->RenderFile(strSourceFile); // 将读取的文件传递给 RenderFilter 解码
m_FilterGraph->SetDisplay Window(m_VideoWindow.GetSafeHwnd());
m_FilterGraph->SetNotify Window(this->GetSafeHwnd()); // 设置显示窗口并通知
```

视频流每次经过 Filter 时,都会调用 Transform 函数进行数据处理,即将 SourceFile 的数据流转换为 OpenCV 可以处理的图像信息,并运用背景差分法对图像进行检测并输出。由于 DirectShow 和 OpenCV 中数据结构不同,但包含相同的数据内容,因此两者之间要完成数据的转换。也就是在 OpenCV 相关函数初始化后,将 DirectShow 所支持的数据结构中的图像数据拷贝到 OpenCV 所支持的图像数据部分,进一步完成车辆检测算法的核心部分。

2.3 基于多帧平均算法的背景差分法

背景差分法是利用当前帧图像与背景图像做差分来提取感兴趣区域的一种常用的运动目标检测算法^[5]。本文基于背景差分法的车辆检测算法思想如下:

1) 使用 DirectShow 获取交通视频中一帧一帧连续的图像,并进行灰度化和平滑处理;

2) 初始背景建立。为了不使初始背景失真,采用多帧图像统计求平均值的方法,避免因第一帧有运动

目标而导致背景检测出现错误的物体团块。即在帧序列号 frameNum<50,使用进行背景加权叠加。当 frameNum = 50时,采用平均值法求得的初始背景;

3) 将当前帧与背景图中值滤波后进行差分运算,同时统计计算其直方图的灰度级;

4) 采用OSTU算法求阈值,即计算从1至 m 的每个灰度级,并计算类内方差和类间方差,具有最大方差的 T 即为该阈值;

5) 更新背景,并提取出感兴趣的区域,并将其二值化;

6) 根据连续帧的二值化图像检测出新进入的车辆,从而完成车辆的检测。

由于交通场景中环境的变化,导致背景图像会随时发生变化,如果直接将当前图像和未更新的背景图像进行差分,会产生大量误差,因此如何建立合适的背景模型是背景差法需要解决的关键问题之一。

本文采用多帧图像统计平均算法^[6]来确定初始背景图像,首先利用统计直方图的方法估计每一个位置处每一种可能像素灰度值出现的次数,出现次数最多的位置即作为背景点

$$N_i(x, y, m) = \begin{cases} N_{i-1}(x, y, m) + 1 & C_i(x, y) = m \\ N_{i-1}(x, y, m) & C_i(x, y) \neq m \end{cases} \quad i = 1, 2, \dots, N \quad (1)$$

式中: $C_i(x, y)$ 表示图像序列中的图像像素点的灰度值; $N_i(x, y, m)$ 表示 $C_i(x, y)$ 对应的背景图像像素点的灰度值; m 为某特定级的灰度值。

该方法原理简单,但运算量较大,对背景变化的反应较慢。为了解决该问题,考虑将某点的灰度范围等分为若干个区间,这样原本统计某一位置处的像素灰度值的次数转变为统计该像素落在各区间段的次数,也就是计算落在每个区间内像素点的灰度值的平均值 μ 和区间的计数统计 S ^[7]。

当前 i 帧在第 n 个区间上的均值 $\mu_{i,n}$ 的表达式可参照多帧平均的更新公式,即

$$B_i(x, y) = \frac{1}{N} \sum_{i=t-N+1}^t C_i(x, y) \quad (2)$$

式中: $B_i(x, y)$ 表示初始背景图像中像素点的灰度值; $C_i(x, y)$ 表示当前图像中像素点的灰度值。

当 $C_i(x, y) \in \left[(n-1) \times \frac{256}{N}, n \times \frac{256}{N} \right]$ 时

$$\mu_{i,n} = (1 - \alpha) \cdot \mu_{i-1,n}(x, y) + \alpha \cdot C_i(x, y) \quad (3)$$

当 $C_i(x, y) \notin \left[(n-1) \times \frac{256}{N}, n \times \frac{256}{N} \right]$ 时

$$\mu_{i,n} = \mu_{i-1,n}(x, y) \quad (4)$$

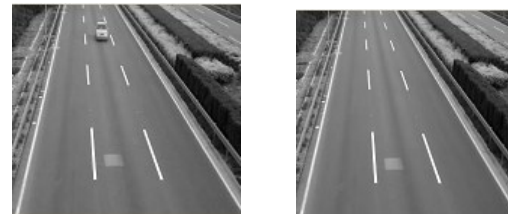
式中: $\mu_{i,n}$ 表示更新后的背景在第 n 个区间上的均值; $\mu_{i-1,n}(x, y)$ 表示当前背景在第 n 个区间上的均值; α 表示权值。

对帧中的每个像素点计算 $\mu_{i,n}$ 和 $S_{i,n}$ 后,比较不同区间段的 $S_{i,n}$,具有最大 $S_{i,n}$ 的区间的 $\mu_{i,n}$ 即为背景。

在OpenCV中提供了完成该功能的函数,其定义^[8]如下所示:

```
Void cvRunningAvg(const CvArr * image,
                  CvArr * acc,
                  double alpha,
                  const CvArr * mask=NULL);
```

其中: image 为输入图像指针; acc 为累加结果指针; alpha 为累加权重; mask 为蒙板图指针。当 mask 为 NULL 时, image 图像中的每一个像素点以 alpha 为权重累加到 acc 中; 若 mask 不为 NULL 时, 只将 mask 中



(a) 原始图像

(b) 背景图像

图1 多帧图像统计平均算法得到的背景图像
Fig.1 Background image gained by the algorithm of multi-frame average

对应像素值为1的image中的像素点的灰度值累加到acc中,这样就完成了不同区间内像素点的累加,之后将其累加值求平均值得到背景图像。

图1为多帧图像统计平均算法得到的背景图像,其中(a)为含有运动目标的原始图像;(b)为由该算法生成的灰度背景图像。从图中可见,背景图像总体保持了一个相对纯净的背景环境,为进一步运动目标的检测奠定了基础。

3 运动目标检测结果分析

本文实验环境如下:

1) 硬件条件:PC机一台(奔4CPU,256M内存,20G硬盘);DeckLink采集卡一块(采用PCI接口)

2) 软件条件:Windows xp系统,VC++6.0,OpenCV 2.0,DirectShow

实验中选取两个不同环境对系统加以评价,将漏检率和准确率^[9]作为指标评价系统的性能。其定义如下:

$$\text{漏检率} = \text{漏检车辆数} / (\text{正确车辆数} + \text{漏检车辆数}) \times 100\%$$

$$\text{准确率} = \text{正确车辆数} / (\text{正确车辆数} + \text{错检车辆数}) \times 100\%$$

检测结果如图2和图3所示。



图2 在场景1下运动车辆检测效果比较

Fig.2 Effects of moving vehicle detection in the scene 1



图3 在场景2下运动车辆检测效果比较

Fig.3 Effects of moving vehicle detection in the scene 2

两个场景下目标检测的准确性如表1所示。

表1 运动目标提取结果统计

Tab.1 Statistic results of moving object extraction

交通场景	正确车辆数/辆	漏检车辆数/辆	错检车辆数/辆	准确率/%	漏检率/%
场景1	1 038	11	2	99.8	1.05
场景2	893	8	1	99.9	0.89

通过实验结果可以看出,利用本算法可以比较完整地将运动目标从背景区域中分离出来,得到比较理想的检测效果;在PC机上每秒钟可处理18帧的图像,提高了算法的实时性和可靠性,但在下雨或光照强烈的场景下,车辆的检测准确率有所下降,还需进一步改进。

4 结论

以城市交通监控场景为背景,检测视频中的运动目标,采用VC++6.0、DirectShow和OpenCV技术设计实现了一个基于多帧平均的背景差分法的车辆检测系统。运动DirectShow对各种视频格式文件编码,实现对流媒体的捕捉、回放和帧操作,使用Opencv库对捕获的图像帧进行灰度化、平滑处理、帧间差分等处理,根据多帧平均的背景差分算法检测运动车辆。同时将该算法应用于不同场景下,进一步验证了该算法的正确性。通过实验结果可知,基于改进的背景差分法的交通检测系统能较好的检测出运动目标,算法的复杂度不高,实时性较好,具有较高的可行性和广泛的应用前景。

参考文献:

- [1] 陆其明. DirectShow开发指南[M]. 北京:清华大学出版社,2003:12-34.
- [2] 陈宁,刘宏刚,张向东,等. 基于DirectShow技术的视频采集方案的实现[J]. 电子科技,2007(2):66-69.
- [3] 姜永成,周正干,任福君,等. 基于OpenCV的移动机器人视频流采集与处理[J]. 机床与液压,2010(8):40-43.
- [4] 潘攀. 基于DirectShow的视频监控采集系统[J]. 现代计算机,2010(8):156-158.
- [5] 张建飞. 基于OpenCV的交通视频运动目标检测与跟踪[J]. 电子测量,2012(1):50-53.
- [6] CHEUNG SEN-CHINGS, CHAND R. Robust techniques for background subtraction in urban traffic video[J]. Proc SPIE, Video Communications and Image Processing, 2004, 53(8):881-892.
- [7] 石红芹,吕方亮. 基于双置乱与奇异值分解的彩色图像水印算法[J]. 华东交通大学学报,2010,114(4):44-47.
- [8] 张学贺,张学东,丁宁. 基于OpenCV的运动目标检测与跟踪[J]. 辽宁科技大学学报,2010(10):490-494.
- [9] 郭怡文,袁飞龙. 基于背景差分的多车道车流量检测系统[J]. 电光与控制,2010,9(17):90-93.

A Research of Moving Object Detection in Complex Traffic Based on OpenCV and DirectShow

Liu Xin¹, Ma Hongfeng^{1,2}, Wang Hongbin¹

(1. Department of Electronic and Information Engineering, Lanzhou Industrial College, Lanzhou 730050, China; 2. School of Electronics and Information Engineering, Lanzhou Jiaotong University, Lanzhou 730070, China)

Abstract: This paper presents an improved algorithm based on automatic detection of moving targets. The background image can be obtained and updated by a multi-frame average, and vehicle outline can be detected. Using VC++6.0 as a developing environment, the algorithm can be programmed by OpenCV and DirectShow library. The results show the improved algorithm can detect moving vehicles correctly in the two different scenes, and it can satisfy the requirement of real-time.

Key words: OpenCV; DirectShow; background difference; multi-frame averaging algorithm