

参赛密码 \_\_\_\_\_  
(由组委会填写)



## “华为杯”第十四届中国研究生 数学建模竞赛

### 题 目      基于监控视频的前景目标提取

#### 摘            要：

随着视频监控的智能化发展,对视频中运动目标的识别提取在智能监控系统中显得愈来愈重要。运动目标检测是计算机视觉研究中非常重要的一部分,它的目的是将用户感兴趣的运动目标准确、完整的从视频序列中提取出来。本文研究了视频监控系统中静态背景和动态背景下运动目标检测技术,当摄像机发生晃动或偏移下的视频去抖动算法,选出包含显著前景目标的视频帧标号的方法,多摄像机不同视角拍摄视频的前景目标提取以及视频异常事件检测算法。

针对问题一:静态背景、摄像头稳定拍摄下的前景目标提取。静态背景下只有前景目标在运动,而背景(如地面、天空、建筑物等)仅存在微小的变化或没有变化。现有的静态背景下运动目标检测算法按照基本原理的不同可以分为三大类:背景减除法、帧间差分法、光流法。此外,ViBe 算法由于效果优于所常用的几种算法,近年来成为研究运动目标提取的新方法。本文分别采用帧差法、平均背景差法、ViBe 算法进行静态背景下的视频前景提取实验,并对实验结果作对比及分析。

针对问题二:动态背景下的前景目标提取。动态背景下的背景和前景目标同时在运动,相对于静态背景来说,这提供了更加丰富的信息量,但是也为运动目标的检测工作带来了更大的挑战。对此,本文分别采用混合高斯模型的运动目标检测算法和优化的 ViBe 算法对动态背景下的视频进行前景目标提取,并对比分析了其实验结果。

针对问题三:摄像头发生晃动或偏移下的前景目标提取。问题一和和问题二分别研究了在静态背景和动态背景下视频前景目标提取的算法,而摄像机在实际拍摄的过程中难免会发

生晃动或偏移，这就造成了拍摄出来的视频会有抖动。因此，为了去除视频的抖动影响，需要根据原视频产生经过运动补偿过的视频序列。这种方法关键就在于怎样利用帧间的信息来得出当前帧的全局运动参数。通过查阅相关资料，Random Sample Consensus (RANSAC) 算法在估计运动参数时有很强的鲁棒性，故本文采用 RANSAC 算法来计算帧的全局运动参数，进而去除视频的抖动，并利用此方法在典型监控视频中验证了其有效性。

针对问题四：选出包含显著前景目标的视频帧标号。前三问解决了前景目标的提取方案以及视频去抖动问题，能有效的将前景目标从静态和动态背景中提取出来。在此基础上，本文利用问题二中提出的改进的 ViBe 算法将 8 组视频的前景目标提取出来，并保存为作为视频帧号标记的输入视频。其中前景目标提取视频序列为二值化序列，前景目标为白色区域，背景为黑色区域，我们利用前景目标面积双阈值将目标帧号和背景帧号标记出来。

针对问题五：近似同一地点不同角度的多个监控视频中前景目标检测与提取。在视频监控系统中，多摄像机的应用可以提供多个视角的信息。因此，在多摄像机视频监控系统中，首先利用近似同一地点在不同高度层上的标志物，计算基于多层的不同视角间的单应性矩阵，然后利用混合高斯模型背景建模，得到多个视角的前景似然图像。最后，通过单应性变换获得目标在不同高度层的定位信息，利用前文所提出的运动目标提取算法进行前景目标的检测和提取。

针对问题六：视频中异常事件检测。群体性异常事件（如人群短时聚集、惊恐逃散和拥挤踩踏等）从图像处理的角度来看，群体中的个体运动存在随机性和无序性，无法提取有序、规律的运动模式；场景中的复杂背景背景动态变化，而且遮挡严重，背景减除法和显著点检测等传统算法都不太适用，故很难对个体进行有效的运动检测。本文在利用光流特征的前提下，建立了事件均值评分机制模型来判断场景中是否发生异常事件。文中首先提取前景目标图像的光流场特征，为了更加突出运动特征对异常事件检测结果的影响，提出光流速度分离的概念，滤除光流场中的低速部分信息，保留高速部分信息，然后根据均值评分模型检测异常事件的发生，最后利用 UMN 数据库中的异常事件视频来进行实验验证异常事件检测的有效性。

本文的特色在于，在问题四中，采用了前景目标双阈值的方法将显著目标和非显著目标的帧号都可以标注出来，可以标记出较小目标出现的帧号。在问题六中，引入了事件均值评分机制来检测异常事件的发生。

最后，本文对所建立的模型给出了评价和改进的方向。

**关键词：**前景目标提取；背景减除法；ViBe 算法；视频去抖算法；前景目标面积双阈值；多视角目标检测；光流特征；均值评分

# 目 录

1 问题重述 .....	5
1.1 问题背景 .....	5
1.2 问题提出 .....	5
2 模型假设 .....	5
3 符号说明 .....	6
4 问题分析 .....	6
5 模型的建立与求解 .....	7
5.1 问题一：静态背景下的前景目标提取 .....	7
5.1.1 问题分析 .....	7
5.1.2 图像预处理 .....	7
5.1.3 常见的运动目标检测算法 .....	8
5.1.4 ViBe 算法 .....	10
5.1.5 算法实验结果比较 .....	12
5.2 问题二：动态背景下的前景目标提取 .....	12
5.2.1 问题分析： .....	12
5.2.2 混合高斯背景建模 .....	13
5.2.3 优化的 ViBe 算法 .....	15
5.2.3.1 样本一致性 (SACON) 背景建模 .....	15
5.2.3.2 形态学处理 .....	17
5.2.3.3 实验结果 .....	18
5.3 问题三：摄像头发生晃动或偏移下的前景目标提取 .....	19
5.3.1 问题分析 .....	19
5.3.2 去抖动算法 .....	19
5.3.3 运动参数估计 .....	19
5.3.4 RANSAC 算法 .....	20
5.3.5 运动参量滤波 .....	20
5.3.6 图像运动补偿 .....	21
5.3.7 实验结果 .....	21
5.4 问题四：标记包含显著前景目标的帧号 .....	22
5.4.1 问题分析 .....	22

5.4.2 模型的建立求解.....	22
5.4.3 实验结果分析.....	22
5.5 问题五：近似同一地点不同角度的多个监控视频中前景目标检测与提取 .....	23
5.5.1 问题分析.....	23
5.5.2 单应性矩阵.....	24
5.5.3 基于不同高度层的目标定位.....	24
5.5.4 实验结果.....	25
5.6 问题六：视频中异常事件检测.....	28
5.6.1 问题分析.....	28
5.6.2 光流场特征提取.....	28
5.6.3 异常事件检测.....	31
5.6.4 异常事件评分方法.....	32
5.6.7 实验结果.....	32
6 模型的评价与改进 .....	33
6.1 模型的评价.....	33
6.2 模型的改进.....	33
7 参考文献 .....	34
8 附录 .....	34
8.1 附录 1.....	34
8.2 附录 2.....	42

# 1 问题重述

## 1.1 问题背景

视频监控是中国安防产业中最为重要的信息获取手段。随着“平安城市”建设的顺利开展，各地普遍安装监控摄像头，利用大范围监控视频的信息，应对安防等领域存在的问题。近年来，中国各省市县乡的摄像头数目呈现井喷式增长，大量企业、部门甚至实现了监控视频的全方位覆盖。如北京、上海、杭州监控摄像头分布密度约分别为 71、158、130 个/平方公里，摄像头数量分别达到 115 万、100 万、40 万，为我们提供了丰富、海量的监控视频信息。

目前，监控视频信息的自动处理与预测在信息科学、计算机视觉、机器学习、模式识别等多个领域中受到极大的关注。因此，有效、快速抽取出监控视频中的前景目标信息，是其中非常重要而基础的问题。这一问题的难度在于，需要有效提取前景目标的视频往往包含静态、动态的背景。因此，此技术已被广泛用于视频目标追踪，城市交通检测，长时场景监测，视频动作捕捉，视频压缩等中，对维护城市交通秩序、提升公安工作效率、保障人民安全等方面具有重要意义。

## 1.2 问题提出

问题一：实验对象是静态背景下的摄像头稳定拍摄 5 秒左右的监控视频，对其构造前景目标提取的数学模型，并详细介绍求解该模型的方法。

问题二：实验对象是动态背景下摄像头稳定拍摄的监控视频，设计有效的前景目标提取方案。

问题三：针对摄像头晃动或偏移状态下拍摄的监控视频，设计有效的提取前景目标方案。

问题四：采用之前所构造的建模方法提取出附件 3 中 8 组视频的显著前景目标，并且记录下包含显著前景目标的视频帧号及在论文中独立成段表示。必须注明前景目标是出现于哪一个视频的哪些帧。

问题五：实验对象是不同角度同时拍摄的近似同一地点的多个监控视频，重复考虑并利用多个角度视频的前景之间的相关性信息，设计方案实现有效监测和提取视频的前景目标。

问题六：利用所提取的前景目标信息，自动判别监控视频中是否有异常事件发生，包括人群短时聚集、人群惊慌逃散、群体规律性变化（如跳舞、列队排练等）、物体爆炸、建筑物倒塌等。其中，可以利用的特征信息包括前景目标奔跑的线性变化形态特征、前景规律性变化的周期性特征等。并且设计出相应的异常事件检测方案。

# 2 模型假设

假设 1：图像在计算和传输的过程中不会失真。

假设 2：图像的灰度是 0~255。

假设 3：相邻像素点是具有相似的时空分布。

假设 4：目标运动场在水平和竖直方向上的速度是连续光滑的。

### 3 符号说明

符号	符号说明
ViBe	视觉背景提取
GMM	混合高斯模型
SACON	样本一致性
RANSAC	随机样本一致性
SAD	绝对差和
GAE	全局异常事件

### 4 问题分析

本文研究了视频监控系统中静态背景和动态背景下运动目标检测技术,当摄像机发生晃动或偏移下的视频去抖动算法,选出包含显著前景目标的视频帧标号的方法,多摄像机不同视角拍摄视频的前景目标提取以及视频异常事件检测算法。

对于问题一:静态背景、摄像头稳定拍摄下的前景目标提取。静态背景下只有前景目标在运动,而背景(如地面、天空、建筑物等)仅存在微小的变化或没有变化。现有的静态背景下运动目标检测算法按照基本原理的不同可以分为三大类:背景减除法、帧间差分法、光流法。此外,ViBe 算法由于效果优于所常用的几种算法,近年来成为研究运动目标提取的新方法。本文分别采用帧差法、平均背景差法、ViBe 法进行静态背景下的视频前景提取实验,并对实验结果作对比及分析。

对于问题二:动态背景下的前景目标提取。本文分别采用混合高斯模型的运动目标检测算法和优化的 ViBe 算法对动态背景下的视频进行前景目标提取,并对比分析了其实验结果。其中,混合高斯模型采用高斯分布描述法表示视频帧中每个像素的灰度值,并通过高斯模型的拟合形成背景图像,将当前帧与背景图像逐像素比较,以此区分图像中前景点和背景点。针对问题一中传统 ViBe 算法忽略时间域上样本的采集,由于背景模型更新过慢产生的鬼影,本文提出一种改进算法,不仅仅加入时间域上的采集,还融合了 SACON 算法中的 TOM 更新方式,并且添加了阴影去除和形态学连通区域分析。

对于问题三:摄像头发生晃动或偏移下的前景目标提取。问题一和和问题二分别研究了在静态背景和动态背景下视频前景目标提取的算法,而摄像机在实际拍摄的过程中难免会发生晃动或偏移,这就造成了拍摄出来的视频会有抖动。视频去抖动的过程就是要根据原视频产生经过运动补偿过的视频序列,这样就可以将不需要的抖动除去。这种方法关键就在于怎样利用帧间的信息来得出当前帧的全局运动参数。通过查阅相关资料,Random Sample Consensus (RANSAC) 算法在估计运动参数时有很强的鲁棒性,故本文采用 RANSAC 算法来计算帧的全局运动参数,进而去除视频的抖动。

对于问题四:选出包含显著前景目标的视频帧标号。本文利用问题二中提出的改进的 ViBe 算法将 8 组视频的前景目标提取出来,并保存为作为视频帧号标记的输入视频,利用目标面积双阈值标记出包含显著目标的帧号。

对于问题五:近似同一地点不同角度的多个监控视频中前景目标检测与提取。在视频监控系统中,多摄像机的应用可以提供多个视角的信息。多摄像机视频监控系统中,首先利用近似同一地点在不同高度层上的标志物,计算基于多层的不同视角间的单应性矩阵,然后利用混合高斯模型背景建模,得到多个视角的前景似然图像。最后,通过单应性变换获得目标在不同高度层的定位信息,利用前文的运动目标提取算法进行前景目标的检测和提取。

对于问题六:视频中异常事件检测。群体性异常事件(如人群短时聚集、惊恐逃散和拥挤踩踏等)从图像处理的角度来看,群体中的个体运动存在随机性和无序性,无法提取有序、规律的运动模式;场景中的复杂背景动态变化,而且遮挡严重,背景减除法和显著点检测等传统算法都不太适用,故很难对个体进行有效的运动检测。本文在利用光流特征的前提下,建立了事件均值评分机制模型来判断场景中是否发生异常事件。根据这个均值评分检测异常

事件的发生，利用 UMN 数据库中的异常事件视频来进行实验验证异常事件检测的有效性。

## 5 模型的建立与求解

### 5.1 问题一：静态背景下的前景目标提取

#### 5.1.1 问题分析

运动目标的提取就是将我们所感兴趣且状态为运动的前景从静止的背景中分离出来进行更深层次的分析。运动目标提取是目标跟踪、特征提取、行为识别这所有后续处理的基础，提取出来的效果好坏会对后面的处理造成直接的影响，所以，运动目标的提取是至关重要的一步。实际监控中，我们需要考虑诸多因素带来，如噪声的影响、光照的变化以及阴影因素的干扰等。静态背景下只有前景目标在运动，而背景（如地面、天空、建筑物等）仅存在微小的变化或没有变化。现有的静态背景下运动目标检测算法按照基本原理的不同可以分为三大类：背景减除法、帧间差分法、光流法。此外，ViBe 算法由于效果优于所常用的几种算法，近年来成为研究运动目标提取的新方法。因此，本文分别采用帧差法、平均背景差法、ViBe 法进行静态背景下的视频前景提取实验，并对实验结果作对比及分析。

#### 5.1.2 图像预处理

运动目标检测的前期处理工作可以称为图像预处理，其主要任务是最大限度的提升检测性能，如提高检测准确性和降低运算量等，图像预处理对运动检测系统具有非常重要的作用。

数字图像在采集的过程中会受到很多的干扰噪声，典型的几种噪声有：

①高斯噪声是从电子电路带来的噪声以及周围环境照明度较低或温度高引起的传感器噪声；

②椒盐噪声和在图像上零散分布的胡椒和盐粉微粒很相像，噪声的产生原因主要是图像分割或域变化引发的；

③加性噪声是图像在传输中引进的信道噪声；

④脉冲噪声是电磁干扰引进的噪声。

给人视觉感很强的孤立像素点或者像素块是上述这些噪声的通常表现形式，它们在数字图像上呈现出极大或极小的极值，最终影响图像的真实性，对图像的处理有极大的干扰，严重影响对运动目标检测的准确性。其中尤其以椒盐噪声和脉冲噪声的影响最大，会使图像产生孤立的前景点，或者是目标镂空等现象。

去噪方法可分为空域法和频域法两种。第一种是空域法，它采用图像模板与图像卷积，从而消除噪声。另外一种为频域法，是通过对图像进行一定变化后采用滤波器，之后再反变换得到去噪的图像。图像中所表现的噪声常见的都是加性随机噪声，通常可通过中值滤波、均值滤波、高斯滤波等滤波方法去除这些噪声，提高信噪比。对于噪声分布比较均匀，而峰值并不是很高的情况下使用均值滤波可以获得较好的去噪效果；对于尖脉冲噪声等类型的噪声使用中值滤波能够取得有较好的噪声滤除效果，还能突出图像的边缘和细节效果；对于高斯白噪声等类型的噪声使用高斯滤波则会有更好的滤除效果。下面就具体介绍均值滤波、中值滤波、高斯滤波去噪的方法。

##### （1）均值滤波

均值滤波属于一种典型的线性滤波算法，而邻域平均法是其使用的主要方法。均值滤波的实现方法是将某一点的值用该点的一个邻域中的像素值的平均值进行替换。

对图像中待处理的某点  $f(x, y)$ ， $S$  为其邻域，选择一个适当的模板，如  $3 \times 3$  或  $5 \times 5$  等，为该领域内的像素点数，用  $g(x, y)$  表示该点的新像素值，则均值滤波法通过公式可表示为：

$$g(x, y) = \frac{1}{M} \sum_{(i, j) \in S} f(x, y) \quad (5.1)$$

通过均值滤波的方法可以把突变的像素点分散到其邻域来实现图像去噪的效果,该方法运算简单,调用时间少,可以平滑图像,并对高斯噪声也能较好的抑制,但是均值滤波无法去除噪声,通过平滑后只能微小的减弱,还会使图像在一定程度上产生失真,图像细节变得模糊。

### (2) 中值滤波

中值滤波是一种非线性的数字滤波算法,它的基本思想是将某一点的邻域中的像素值的中值作为该点的值,使其与附近的像素值更加接近,也就是更靠近真实值,以达到抑制干扰噪声点的目的<sup>[2]</sup>。

对图像中邻域  $S$  为的待处理某点选择一个适当的模板,用  $g(x, y)$  表示的新像素值,则中值滤波法通过公式可表示为:

$$y = \begin{cases} x_{\frac{N+1}{2}} & N \text{ 为奇数} \\ \frac{1}{2} \left( x_{\frac{N}{2}} + x_{\frac{N+1}{2}} \right) & N \text{ 为偶数} \end{cases} \quad (5.2)$$

中值滤波器是一种很常用的去噪方法,原因在于针对一些类型的随机噪声,它可以有效的消除噪声,比一般的线性滤波器的去噪效果优异且处理脉冲噪声等颗粒噪声非常有效。

如果只是简单的改善图像的噪声干扰,可以使用均值滤波等线性滤波去噪方法,但这样会难以避免的使图像细节在一定程度上被丢失使图像失真乃至难以辨认。而中值滤波是一种极有效的非线性去噪方法,可以有效的去除大部分噪声干扰的同时能较好的保持图像细节,同时运算速度较快,可满足系统的实时要求。

### (3) 高斯滤波

高斯滤波是一种线性平滑滤波器,它是通过高斯函数的形状来选择权值,其实就是对整个图像进行加权平均运算的过程。图像中每个像素点的值和它邻域内的其它像素值,经过加权平均运算后可以得到该像素点的最后值。它的基本原理是利用一个适当的模板去对图像中的每一个像素进行扫描,通过模板来获得周围区域内各像素的加权平均灰度值,然后替换掉该模板的中心位置像素点的值<sup>[2]</sup>。

一维零均值高斯函数如下:

$$G(x) = e^{-\frac{x^2}{2\sigma^2}} \quad (5.3)$$

式中  $\sigma$  值的大小决定了高斯滤波的平滑程度,当  $\sigma$  变大时,频带则随之变得越宽,在对噪声滤波时的平滑度也随之变得更好。但是当  $\sigma$  过大时,图像平滑度又会过度,从而造成图像变得模糊。

高斯滤波对抑制高斯噪声等噪声非常有效,去除高频信号的同时又保留有用的信号。

由于高斯函数是单值函数,及其固有的旋转对称性,使高斯滤波对边缘滤波效果极佳。高斯函数还具有可分离性的特性,也就是它的运算量在随模板变化时成线性增长,也就保证实时性得到有效保障。但是高斯滤波存在着一些弊端,主要体现在对椒盐噪声、脉冲噪声等平滑效果不好。

本文所采用的图像预处理方法是中值滤波。

## 5.1.3 常见的运动目标检测算法

运动目标检测算法的根本在于运动目标的检测。实现运动物体从背景中清晰、完整及快速的检测是目标检测算法的目的。运动目标的正确检测对后续的目标分类、描述及跟踪产生非常重要的影响。

目标检测算法主要包括帧间差分法、光流法以及背景减除分法。

### (1) 帧间差分法

帧间差分法是一种简单易行的运动目标检测算法,也称为时域差分法,根据所采用帧数的不同又分为双帧差分法和三帧差分法。前者的基本原理是:选取前一帧图像和当前帧图像,对它们进行差分运算,得到差分图像,然后对该图像进行二值化操作获得目标前景块,最后



进行目标提取。具体流程如 5.1 图所示。

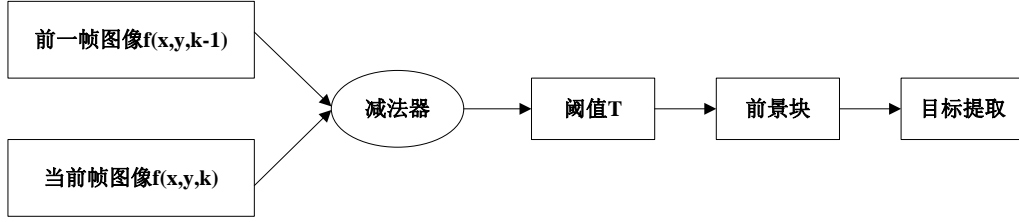


图 5.1 帧间差分法

对于分辨率为  $m \times n$  的图像序列，用  $f(x, y, k-1)$  和  $f(x, y, k)$  分别表示第  $k-1$  帧和第  $k$  帧图像及各自  $(x, y)$  点的像素值，则差分结果  $D(x, y, k)$  为：

$$D(x, y, k) = \begin{cases} 1 & |f(x, y, k) - f(x, y, k-1)| > T \\ 0 & |f(x, y, k) - f(x, y, k-1)| \leq T \end{cases} \quad (5.4)$$

其中  $T$  为二值化阈值，作用是区分前景像素点和背景像素点，当  $D(x, y, k)$  大于设定好的阈值时，即为前景像素点，此时将二值化图像中相应位置的像素点值置为白色；当  $D(x, y, k)$  小于设定好的阈值时，此时将二值化图像中相应位置的像素点值置为黑色，这样就得到了一幅只有黑白像素点的二值化差分图。

双帧差分法的缺点：存在着“空洞”和“双影”效应，运动目标内部大部分地方是空洞的，且目标的边缘比较粗，好像前后两帧前景目标的边缘重叠一样。

三帧差分法可以克服这些缺点，其基本思想是：首先从序列图像中获取连续三帧图像，分别对相邻两帧进行差分运算并进行二值化操作，得到两幅差分图像，然后对这两幅图像进行逻辑与的运算，实现对运动目标的提取<sup>[2]</sup>。

假设连续三帧视频图像分别为  $f(x, y, k-1)$ 、 $f(x, y, k)$  和  $f(x, y, k+1)$ ，则三帧差分法流程如图 5.2 所示：

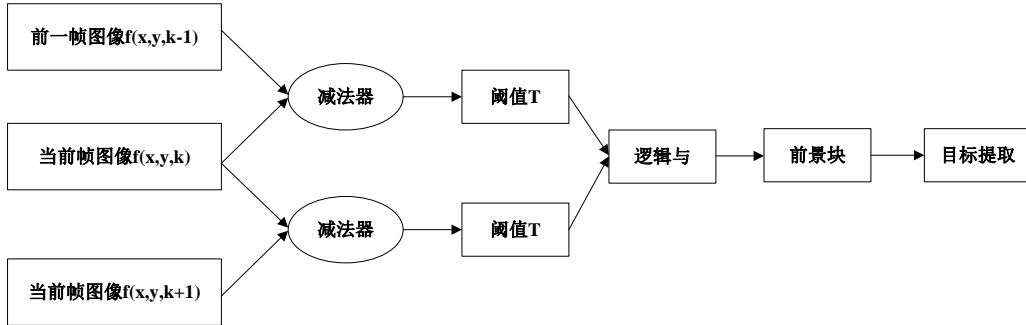


图 5.2 三帧差分法

帧间差分法的优点：算法计算量小，实时性高，简单且容易实现，与背景差分法相比，不需要建立和更新背景模型，只需要相邻两帧就可以提取出运动目标。同时因为两帧图像之间的时间间隔较小，监控场景内的光线等变化因素对算法的影响不大，能够很好地适应动态变化的场景。帧间差分法的缺点也很多，这种方法只能从图像帧序列中提取出运动目标的轮廓，对阈值的选取较为严格，而且由于相邻两帧时间间隔较短，容易产生且容易产生“双影”效应，并且提取出的运动目标内部是空洞的。尽管三帧差分法对其进行了优化，但提取的依然是目标的边界。同时，当监控场景内的目标过多时，提取的目标轮廓会被粘合在一起，且随着帧数的增加，这种现象愈发严重。此外，对于运动速度缓慢的目标，两种方法都显得那么无能为力。

## (2) 光流法

光流法是利用像素点在时域上的强度变化和帧间相关性来计算相邻帧之间的物体运动信息，从而进行运动目标的提取的一种方法。基本原理是：根据运动和光流之间的紧密联系，用递归的方式初始化目标块的特征，然后通过反馈机制实时的调整参数。在用光流法进行目标的检测和跟踪时，首先要选取一些具备明显运动特征的点，并在跟踪的过程中加入一些新的点，然后通过反馈机制纠正上一帧中存在的错误跟踪结果，并将这些错误的点去除，这样就可以对运动目标进行准确的检测和跟踪<sup>[2]</sup>。

光流法适用于动态背景和静态背景，而且用光流法提取出的运动目标同时还保留着其三维结构的信息。此外，与背景差分法相比，当用光流法进行监控场景内的运动目标检测时，不需要预先知道监控场景内的任何信息。但是当发生多光源照射时和目标被遮挡时，会严重干扰光流场的分布，从而导致错误的计算，影响着目标检测的准确性。此外，光流法的计算量较为庞大，实时处理性能较差。

### (3) 背景减除法

背景减除法是静态背景下非常重要的一种目标检测方法，目前已有十几种之多，包括单高斯模型、混合高斯模型、自组织背景检测、码本方法、样本一致性背景建模、ViBe 及其改进算法、PBAS、基于颜色信息的背景建模、统计平均法、中值滤波法、本征背景法、核密度估计法、SuBSENSE 算法等<sup>[3]</sup>。

背景减除法的基本思想和帧间差分法异曲同工，但是它需要预先获取监控场景内干净的背景图像，即建立一个良好的背景模型并实时进行更新，然后将原始视频序列帧与获得的背景图像相减，得到一副差分图像，把该图像进行二值化操作，最终提取出运动目标，其流程如图 5.3 所示：

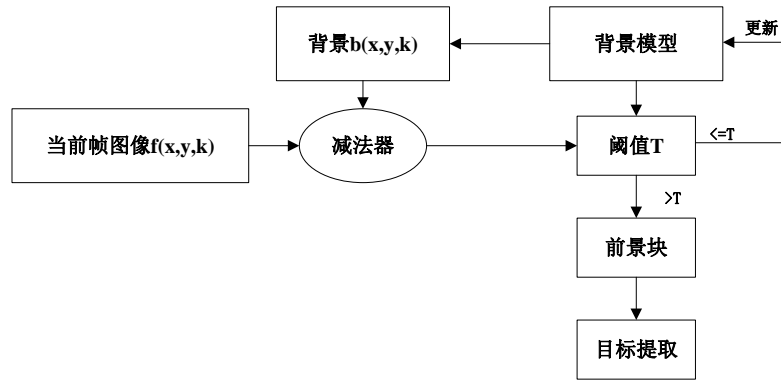


图 5.3 背景减除法

假设视频序列图像中当前帧为  $f(x, y, k)$ ，已获取背景图像为  $b(x, y, k)$ ，这样得出差分结果  $D(x, y, k)$  为：

$$D(x, y, k) = |f(x, y, k) - b(x, y, k)| \quad (5.5)$$

比较  $D(x, y, k)$  和阈值  $T$  的大小，如果  $D(x, y, k)$  中当前像素值大于设定好的阈值  $T$ ，那么可以认为该像素为前景像素；相反，如果  $D(x, y, k)$  中当前像素值小于设定好的阈值  $T$ ，那么该像素为背景像素，如下式 (5.6) 所示：

$$R(x, y, k) = \begin{cases} 1 & D(x, y, k) > T \\ 0 & D(x, y, k) \leq T \end{cases} \quad (5.6)$$

其中， $R(x, y, k)$  值为 0 则为背景像素， $R(x, y, k)$  值为 1 则为前景像素。

背景差分法算法的优点是简单，运算快且容易实现，相比于帧间差分法，背景减除法不受提取目标“空洞”的影响，其可以提取出相对完整的运动前景，其关键在于背景模型的建立，背景模型的好坏直接影响着后续运动目标的检测，所以如何利用视频序列的前几帧快速准确的建立一个干净的背景模型成为背景差分法的中中之重。目前常见的背景建模方法有高斯混合法、均值估计法以及核密度估计法等等。

背景减除法有很多优点，相比于帧间差分法，它可以准确的检测出视频序列帧中的运动目标，而且可以提取出较为完整的信息。此外，该算法计算复杂度小，运行速度快且易实现，通常应用于静态背景下的视频监控，不适用于复杂动态的场景。

#### 5.1.4 ViBe 算法

视觉背景提取法 (Visual Background Extractor, ViBe) 算法是由 Barnich<sup>[4]</sup>提出的一种像素级的前景检测方法，该算法主要创新之处在于背景模型的更新策略，需要更新的像素样本和邻域像素都是随机选择的。

首先进行背景模型的初始化，以上介绍的背景减除法都需要 50~100 帧的视频序列来完

成背景模型的学习过程，而 ViBe 算法仅仅需要一帧即可完成背景模型的建立。对于每个像素来说，利用相邻像素具有相近像素值的特性，从邻域点中随机地选取该像素的模型样本。ViBe 算法的优势在于不仅简化了背景建模过程，而且对于背景突变的情况具有一定的适应性，当检测到背景突然发生明显的变化时，舍弃原始模型，并利用变化后的首帧图像来重新建立背景模型。但是，该算法也存在着一定的缺陷，由于视频首帧可能有运动着的物体，当把它们的像素初始化为样本集后，容易引入鬼影（Ghost）区域。

具体过程是：

#### Step1: 背景建模以及像素点分类

对于一个像素点  $x$ ，在它的邻域点中随机地选择  $N$  个作为其背景样本值，即样本集  $M(x) = (V_1, V_2, \dots, V_N)$ ， $N$  为样本集的大小。从第二帧开始进行前景检测，将当前帧中的像素与其对应的样本进行比较。ViBe 算法求取以当前像素  $x$  为中心， $R$  为半径的区域  $S_R(v(x))$  与  $M(x)$  的交集，如图 5.4 所示，将交集的元素个数与一个给定的阈值  $\#min$  进行比较，若比该阈值大，则判定该像素属于背景，否则为前景。

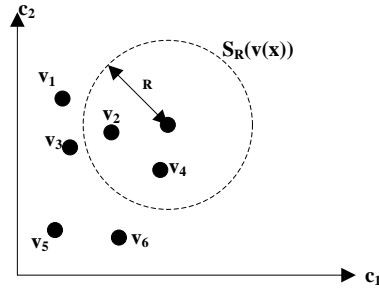


图 5.4 ViBe 背景模型

显然，模型的准确性完全由  $R$  以及  $\#min$  来决定，而算法相应的敏感度由  $\#min/N$  来决定。

#### Step2: 初始化模型

ViBe 算法是由第一帧视频帧来对背景模型进行初始化的，它引入邻域模型来建立每个像素点的背景样本集。背景样本集  $M(x)$  的建立依赖于八邻域  $N_G(x)$ （如图 5.5 所示），具体方法是随机抽取  $N$  个样本值组成样本集。

$V_x(1)$	$V_x(2)$	$V_x(3)$
$V_x(4)$	$V_x$	$V_x(5)$
$V_x(6)$	$V_x(7)$	$V_x(8)$

图 5.5 八邻域模型

这种背景模型的初始化过程简单，前景目标提取速度快，而且算法对于图像噪声以及细小物体的干扰反应灵敏，但缺点就是较容易引入鬼影。

#### Step3: 模型更新

背景模型的更新过程能够使背景模型不断地适应背景的变化，ViBe 算法进行背景更新时随机地选取一个样本值进行更新，保证了每个样本值的生命周期以指数形式单调递减，即一个样本值从时刻  $t$  经过时间  $dt$  仍然被保留下来的概率是：

$$P(t, t + dt) = \left(\frac{N-1}{N}\right)^{(t+dt)-t} = e^{-\ln(\frac{N}{N-1})dt} \quad (5.7)$$

其中  $N$  为样本的个数，满足无记忆性。具体的更新方法是：按照一定的更新率进行背景模型的更新，而非所有数据全部都更新。当一个像素被判定为背景时，根据像素的空间传播特性使背景模型向外扩散，它有  $1/\Phi$  的概率去更新自身的模型，与此同时也有  $1/\Phi$  的概率去更新其邻域点的样本值，这样有利于快速地识别 Ghost 区域；而当一个像素被判定为前景点时，需要对其进行计数，只有当计数到达临界值时才将其更新为背景，并且同样有  $1/\Phi$  的概率去更新其自身的模型样本值。

ViBe 算法本身并没有多少计算过程，算法简单，模型更新量小，并具有较好的抗噪能











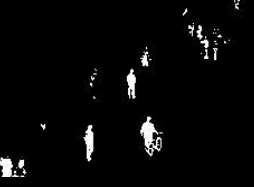





力。与之前的三种算法相比，性能效果上有很强的实用性。然而采用固定的速度更新模型对于运动缓慢或者较为快速的情况都有一定的局限性。前景目标运动缓慢时，前景目标会在短时间内被认为背景点；而当前景目标运动快速时，背景容易来不及更新而形成残影。

### 5.1.5 算法实验结果比较

- (1) 硬件环境：Intel 酷睿 i5-3210M 2.50GHz 处理器、8G 内存的 PC 机上。
- (2) 软件环境：Win7 64bit 系统下的 Matlab2012b 版本。

本文提取静态背景下视频前景目标的实验结果对比如表 5.1 所示：

表 5.1 各算法的实验结果对比

视频名称	原图	帧差法	背景减除法	
			平均背景差法	ViBe 算法
hall				
office				
pedestrian				
smoke				

由以上实验结果对比可知，帧差法提取的前景目标内部存在空洞，能基本完成静态背景下的前景目标，提取效果一般；平均背景差法是对整个视频序列像素值作平均，从而得到背景模型，再利用每帧图像减去背景模型得到前景目标，提取前景目标的效果一般；采用 ViBe 法时，由于前景目标运动缓慢，造成前景目标部分被认为是背景导致目标缺失，且存在较多的噪声，提取前景目标的效果一般。综上，仅仅采取经典的运动目标检测算法来进行前景提取，效果不是很理想。

## 5.2 问题二：动态背景下的前景目标提取

### 5.2.1 问题分析：

问题一中背景减除法通常可以分为 4 个过程，即预处理、背景建模、前景检测和后处理，其中背景建模过程是背景减除法的关键环节，所以背景减除法又称为背景建模技术。

混合高斯模型采用高斯分布描述法表示视频帧中每个像素的灰度值，并通过高斯模型的拟合形成背景图像，将当前帧与背景图像逐像素比较，以此区分图像中前景点和背景点。该方法能够抑制噪声对前景检测的影响，但是当背景中有树枝、水波等多模态形式的变化时，可能出现误检现象。

动态背景下的背景和前景目标同时在运动，相对于静态背景来说，这提供了更加丰富的信息量，但是也为运动目标的检测工作带来了更大的挑战。对此提出了混合高斯模型的运动目标检测算法和优化的 ViBe 算法。

### 5.2.2 混合高斯背景建模

2000 年，Tauffer 与 Grimson 提出了混合高斯分布的思想，即用多个高斯函数去建立图像中每个像素点的状态模型。对像素建模时采用的高斯函数越多，则建立的模型越精确，从而反应的细节状态也越多，但同时也增大计算量，且其更新速率低，对复杂背景环境下的处理效果不好<sup>[5]</sup>。一般情况下采用 3 到 5 个高斯函数，基于混合高斯模型的运动目标检测算法流程如图 5.6 所示：

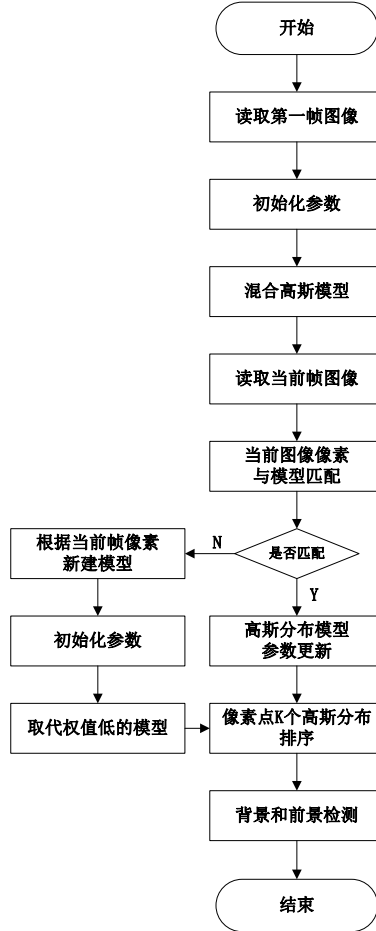


图 5.6 混合高斯模型运动目标检测流程图

在连续帧图像中，假设一个像素点的亮度值当做是一个随机变化的过程，即图像中  $(x_0, y_0)$  点的像素的亮度值序列：

$$\{X_1, X_2, \dots, X_t\} = \{I(x_0, y_0, i) : 1 \leq i \leq t\} \quad (5.8)$$

式中， $I(x_0, y_0, i)$  表示视频中连续的图像序列，表示像素点  $(x_0, y_0)$  在时刻  $i$  的亮度值，也可以利用其他特征，本节以亮度特征为例进行介绍。

单分布高斯背景模型认为，一个图像背景部分的像素值应该满足高斯分布，即在背景图像  $B$  中，像素点  $I(x, y)$  的值满足：

$$B_t(x, y) \sim N(\mu, \sigma^2) \quad (5.9)$$

混合高斯模型则为图像中的每个像素点构建  $K$  个高斯分布，那么在  $t$  时刻对于图像亮度序列  $\{X_1, X_2, \dots, X_t\}$  中  $X_t$  在该模型下的概率分布可用下式 (5.10) 来描述：

$$P(X_t) = \sum_{i=1}^K \omega_{i,t} \eta(x_t, \mu_{i,t}, \sigma_{i,t}) \quad (5.10)$$

式中， $K$  为混合高斯模型中的高斯分布个数， $\omega_{i,t}$  为  $t$  时刻第  $K$  个高斯分布的权值，且

$\sum_{i=1}^K \omega_{i,t} = 1$ ， $\eta(x_t, \mu_{i,t}, \sigma_{i,t})$  表示均值与协方差矩阵分别为  $\mu_{i,t}$ ， $\sigma_{i,t}$  的第  $i$  个高斯分布函数，且有以下式：

$$\eta(x_i, \mu, \sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} |\sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(x_i - \mu)^T \sigma^{-1} (x_i - \mu)} \quad (5.11)$$

其中， $n$  为  $x_t$  的维数。

当下一帧图像到来时，则获得新的像素值  $x_{t+1}$ ，然后根据新像素值对高斯模型进行更新，最后根据更新模型结果判断该像素点是不是前景像素点，整体步骤如下：

**Step1: 初始化背景模型**

**Step2: 将像素值与高斯函数模型匹配**

将像素值与混合高斯模型中的每个高斯函数的均值进行比较，直到找到第一个匹配的高斯函数为止，其匹配依据：

$$M_{m,t} = \begin{cases} 1 & |X_t - \mu_{m,t}| \leq D\sigma_{m,t} \\ 0 & |X_t - \mu_{m,t}| > D\sigma_{m,t} \end{cases} \quad (5.12)$$

式中， $D$  为常数参数，参数  $D$  的取值决定于场景中噪声，通常取 2~3； $M_{m,t}$  为模型匹配算子。若取值为 1，则当前像素与背景模型匹配，否则不匹配。

其匹配结果有如下两种情况：

①有高斯函数与该像素值匹配

当对于一个新的像素值，存在高斯函数与之相匹配时，则更新模型权值：

$$\omega_{i,t+1} = (1 - \alpha)\omega_{i,t} + \alpha M_{i,t} \quad (5.13)$$

式中， $\alpha$  是自定义学习更新速率，且  $0 \leq \alpha \leq 1$ 。

权值更新后，还需对其进行归一化处理：

$$\omega_{i,t} = \frac{\omega_{i,t}}{\sum_{i=1}^K \omega_{i,t}} \quad (5.14)$$

使其满足  $\sum_{i=1}^K \omega_{i,t} = 1$  即可。

对于高斯模型均值与方差的更新如下：

$$\mu_{m,t+1} = (1 - \beta)\mu_{m,t} + \beta X_t \quad (5.15)$$

式中， $\beta$  是参数的学习因子。

②无高斯函数与该像素匹配

在没有高斯函数与该像素匹配的情况下，其均值和方差不做改变，权重按照公式进行更新：

$$\omega_{i,t+1} = (1 - \alpha)\omega_{i,t} \quad (5.16)$$

找出所有高斯函数中权重最小（即最不可能）的高斯函数，并用一个新的高斯函数去代替该高斯函数。新的离斯函数的均值为当前像素的像素值，方差给予一个较大值，权重给予一个较小值。

**Step3: 对权重进行归一化处理，然后对所有高斯函数进行排序**

排序是按优先级  $\varphi_{i,t} = \omega_{i,t} / |\sigma_{i,t}|$  由大到小排列，权重值大，方差小的高斯函数优先级高，排在前面，权重值小的排在后面。

#### Step4: 前景分割

一幅图像可以被分为运动前景和背景两部分，前者表示图像上发生运动的部分，后者则为静止不变的部分，当新输入图像的前景与背景模型建立后，一般认为权值较大，方差较小的高斯模型对应的像素点是背景模型的像素点，相反情况下，由于运动目标出现的概率比背景低，那么，可以认为权值较小的模型对应的像素为前景像素。

当在  $t$  时刻，排序在前  $B$  个高斯分布权值求和，当权值和大于预设阈值  $T$  时，由这  $B$  个分布构成背景模型，即：

$$B = \arg \min_B \left( \sum_{i=1}^B \omega_{i,t} > T \right) \quad (5.17)$$

式中， $T$  为背景模型的权重阈值，权重小于它的为背景。因此， $T$  的选择是影响检测效果的关键因素。本文  $\alpha$  取 0.01， $D$  取 2.5， $T$  取 0.25，对一般场景下基于混合高斯背景更新模型的目标检测算法进行了实验验证，其实验效果在下文展示。

### 5.2.3 优化的 ViBe 算法

针对问题一中传统 ViBe 算法忽略时间域上样本的采集，由于背景模型更新过慢产生的鬼影，本文提出一种改进算法，不仅仅加入时间域上的采集，还融合了 SACON 算法中的 TOM 更新方式，并且添加了阴影去除和形态学连通区域分析。

#### 5.2.3.1 样本一致性 (SACON) 背景建模

在概率论与数理统计中，随着样本容量的增加，估计量会与参数的偏差越来越小，这种性质被称之为样本的一致性 (Sample Consensus, SACON) 或者相合性。Wang<sup>[6]</sup> 等人正是利用样本这一性质提出了基于样本一致性的运动目标检测算法。它是基于数据统计而建立的，为每一个像素建立一个样本集，然后对样本进行一致性计算，从而得到背景模型。该算法运算复杂度低，鲁棒性强，检测效果好，十分值得继续研究。其流程如图 5.7 所示。

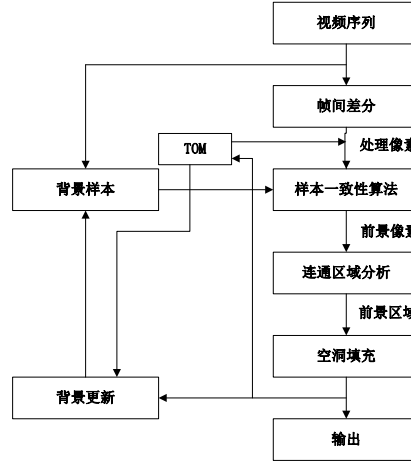


图 5.7 SACON 算法框架

具体步骤如下：

##### Step1: 背景建模

样本一致性算法所建的背景模型既简单又直接，将某个像素的背景模型简单地用  $C$  进行表示，其中  $C = (c_1, c_2, \dots, c_N)$ ，然后直接取视频序列的前  $N$  帧中对应像素的  $N$  个值进行背景模型填充即可。

##### Step2: 前景检测

该算法的前景检测阶段主要包括帧间差分、样本一致性算法、后处理这么几个步骤。下面对这三个部分进行分别介绍。

##### (1) 邻域帧间差分

算法一开始就采用相邻帧间差分法来获得可能的前景像素，这其中或许有误检的情况，

例如将噪声点或者背景点错误的判定为了前景点，但不会对结果产生影响，因为此部分主要是为了减少后续的运算量，加快运算速度。

### (2) 样本一致性算法

当输入新的像素时，将此像素与其对应的背景模型进行比较，从而判断新像素是否满足背景样本的一致性，其中判断公式如下式 (5.18) 和 (5.19)：

$$\Gamma_i^c(m) = \begin{cases} 1 & |x_i^c - x_t^c| \leq T_r \\ 0 & |x_i^c - x_t^c| > T_r \end{cases} \quad (5.18)$$

$$B_t(m) = \begin{cases} 1 & \sum_{i=1}^N \Gamma_i^c(m) \geq T_n \quad \forall c \in \{c_1, c_2, \dots, c_k\} \\ 0 & \sum_{i=1}^N \Gamma_i^c(m) < T_n \quad \forall c \in \{c_1, c_2, \dots, c_k\} \end{cases} \quad (5.19)$$

其中， $T_r$  和  $T_n$  是两个阈值，第一个公式计算新像素与背景模型样本距离，并判断距离是否接近，第二个公式统计距离接近的样本数目，如果距离接近的样本数目总和大于预先设定的阈值  $T_n$  时，则将新像素判断为背景，即  $B_t(m) = 1$ 。

### (3) 后处理

将提取出来的运动目标进行后处理，主要包含以下两个操作：

一是阴影去除，采用下式来验证阴影，并去除阴影区域，公式 (5.20) 如下：

$$\begin{cases} |x_t^c - x_b^c| \leq T_r \quad \forall c \in \{r, g\} \\ \beta \leq \frac{x_t^c}{x_b^c} \leq \gamma \quad c \in \{I\} \end{cases} \quad (5.20)$$

其中， $\{r, g, I\}$  是归一化的色彩空间， $r = \frac{R}{R+G+B}$ ， $g = \frac{G}{R+G+B}$ ， $I = \frac{R+G+B}{3}$ 。

二是连通区域分析和空洞填充，采用的主要方法是形态学处理。

### Step3: 背景更新

为了解决由背景移动或前景停滞所造成的背景变化问题，SACON 算法引入 TOM (Time of Map) 方法进行更新，从而将这种变化很快的融入到背景中去，具体的更新方法分为像素级别和图块级别两种。

#### (1) 像素级别

TOM 计数矩阵累计了一个像素点被连续判定成前景的次数，它的表达式 (5.21) 如下：

$$\begin{cases} TOM_t(m) = TOM_{t-1}(m) + 1 & B_t(m) = 0 \\ TOM_t(m) = 0 & B_t(m) \neq 0 \end{cases} \quad (5.21)$$

当  $B_t(m) = 0$ ，意思是位于  $m$  处的像素点被判成前景点，则给 TOM 值累加 1。持续对 TOM 进行累加，若 TOM 值大于了预先设定的一个阈值，则将这个像素点改判为背景，并将 TOM 值清成 0。

#### (2) 图块级别

图块级别的更新同样依靠 TOM 矩阵来计数，其计算公式 (5.22) 如下：

$$\begin{cases} TOM_t(m') = TOM_{t-1}(m') + 1 & \Omega \text{ 为常数} \\ TOM_t(m') = 0 & \Omega \text{ 不为常数} \end{cases} \quad (5.22)$$

假定目标是一个图块，若其是静止状态，则将该目标内的全部像素的 TOM 值均累加 1；若其是运动状态，则将该目标内的全部像素的 TOM 值均清成 0；若目标的 TOM 值大于预先设定的阈值时，则将该目标内的全部像素改判为背景像素。假设一个目标既包含运动的部分，又包含静止的部分，静止的部分会持续被判定为前景，从而会被像素级别 TOM 更新为背景；可是运动的部分一会儿被当作前景，一会儿时又被当成背景，而且不会被像素级别



*TOM* 更新成背景；最终造成的结果是一个目标被分割，部分被检测为前景，部分被检测为背景，这与现实不符，不合常理。

通过比较，本文采用了像素级别 *TOM* 的更新方式。

### 5.2.3.2 形态学处理

数学形态学<sup>[9]</sup>是用特定形状的结构元素对原始图像进行度量和提取以达到分析及识别之目的。所谓结构元素是指用来处理原始图像的比原图像小的一些图像，原图像和结构元素的关系类似于滤波中图像和模板的关系。数学形态学简称形态学，其常用操作有四种：腐蚀、膨胀、开运算和闭运算。

腐蚀和膨胀是形态学中最基础的两种运算，在此基础上可以推导和组合出多种不同的运算。

#### ①腐蚀运算

腐蚀运算的符号为  $\ominus$ ，假设用  $B$  来腐蚀  $A$  则可以表示成式 (5.23) 所示的形式：

$$A \ominus B = \{x | (B)_x \subseteq A\} \quad (5.23)$$

其中， $A$  为原始图像， $B$  为结构元素， $(B)_x$  表示将结构元素  $B$  平移  $x$ ，腐蚀运算可以理解对结构元素  $B$  做平移后  $B$  全部包含在  $A$  中时的  $B$  的原点位置的集合。可用图 5.8 对腐蚀运算做进一步解读：其中，(a)为原始图形，(b)为结构元素，中间白色的十字形表示原点，(c)中深色区域为腐蚀之后的结果（为了对比保留原始区域的大小）。

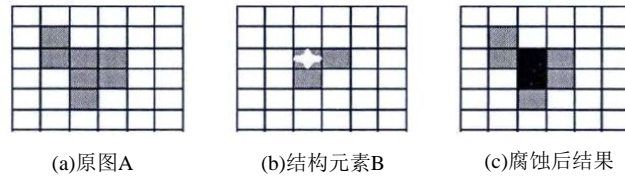


图 5.8 腐蚀过程图示

分析以上结果不难发现：腐蚀能够使物体的边界变小，且变小的结果与结构元素密切相关。当  $A$  大于  $B$  时，腐蚀后  $A$  会小“一圈”；当  $A$  小于  $B$  时，腐蚀后  $A$  会消失不见；当  $A$  中有部分区域小于结构元素  $B$ （如毛刺、小凸起、狭窄连通），则腐蚀后该部分则会被断裂成两部分。

#### ②膨胀运算

膨胀是与腐蚀相对的另一种运算，膨胀的运算符号为  $\oplus$ ，假设用  $B$  来膨胀  $A$ ，则可以表示为如式 (5.24) 所示的形式：

$$A \oplus B = \{\hat{(B)}_x \cap A = \emptyset\} \quad (5.24)$$

其中， $A$  和  $B$  同上， $\hat{(B)}_x$  表示对结构元素  $B$  做关于原点的映射然后再平移  $x$ ，所以式 (5.24) 进行膨胀的结果为对  $B$  做关于原点的对称后在图像  $A$  内进行移动，当二者至少有一个交点时的  $B$  的原点位置的集合。可以用图 5.9 来进一步解读：

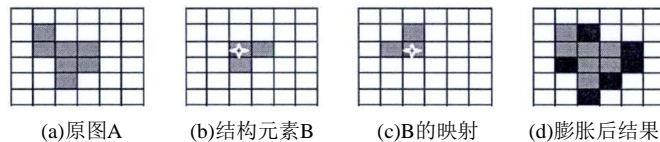


图 5.9 膨胀过程图示

其中，(a)和(b)同上，(c)为  $B$  关于原点的映射，(d)为原点从左上角开始依次对(c)进行平移后与  $A$  交集不为空时的所有原点的集合，即为膨胀后的结果（浅色区域加深色区域）。从图中可以看出，与腐蚀刚好相反，进行膨胀操作后图形  $A$  的边界扩大了，但扩大的具体大小因结构元素而异，故膨胀运算常用于桥接图像中断裂的部分。

对腐蚀和膨胀两种基础运算进行组合便可得到另外两种重要的运算：开运算和闭运算。

### ③开运算

开运算如式 (5.25) 所示, 先用结构元素  $B$  对  $A$  进行腐蚀, 然后再用同一结构元素对腐蚀结果进行膨胀。

$$A \circ B = (A \ominus B) \oplus B \quad (5.25)$$

### ④闭运算

闭运算则正好与开运算相反, 如式 (5.26) 所示, 先膨胀后腐蚀, 两次结构元素相同。


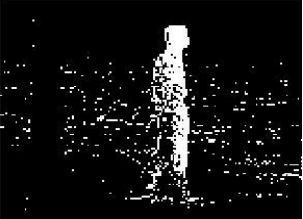









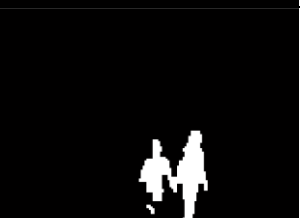
$$A \bullet B = (A \oplus B) \ominus B \quad (5.26)$$

一般来说, 开运算能够使图像变得光滑, 断开比结构元素小的狭窄连接, 消除掉比结构元素小的毛刺、凸起, 同时保证在整体上不产生几何失真; 闭运算也能使图像变得光滑, 但是与开运算相反, 闭运算能够弥合狭窄的间断, 填充小的孔洞, 同时也能保证不产生全局几何失真。本文采用开运算和闭运算相结合的方式去去除噪声。

### 5.2.3.3 实验结果

本文提取动态背景下视频的前景目标的实验结果对比如表 5.2 所示:

表 5.2 算法的实验结果对比

视频名称	帧号	原图	混合高斯背景建模	优化的 ViBe 算法
WaterSurface	46			
Campus	217			
Curtain	1793			
Fountain	191			

由以上实验结果对比可知, 混合高斯背景建模方法与优化的 ViBe 算法比较, 前者提取的前景目标存在较多噪声, 而且目标轮廓没有后者清晰。由此发现, 优化的 ViBe 算法具有较好的抗干扰能力, 可以有效提取动态背景视频下的前景目标。

### 5.3 问题三：摄像头发生晃动或偏移下的前景目标提取

#### 5.3.1 问题分析

问题一和和问题二分别研究了在静态背景和动态背景下视频前景目标提取的算法，而摄像机在实际拍摄的过程中难免会发生晃动或偏移，这就造成了拍摄出来的视频会有抖动。视频去抖动的过程就是要根据原视频产生经过运动补偿过的视频序列，这样就可以将不需要的抖动除去。视频去抖动一般采用的方法是利用相邻帧之间的信息得出当前帧的一个全局的运动参数，通过对此运动参数进行滤波就可以去掉抖动。这种方法关键就在于怎样利用帧间的信息来得出当前帧的全局运动参数。通过查阅相关资料，Random Sample Consensus (RANSAC) 算法在估计运动参数时有很强的鲁棒性，故本文采用 RANSAC 算法来计算帧的全局运动参数，进而去除视频的抖动。

#### 5.3.2 去抖动算法

去抖动算法主要由三部分组成，如图 5.10 所示。



图 5.10 去抖动算法流程图

具体的步骤如下：

##### Step1: 块匹配

每得到一帧图像，先把它和前一帧图像做块匹配。计算块匹配时可以采用 SSD 或 SAD。经过块匹配，得到当前图像的一个比较稠密的运动向量场。

##### Step2: 计算全局运动参数

要用这个运动向量集估计当前帧图像的全局运动参数。视频有局部运动或存在噪声，采用 RANSAC 算法对该运动向量集进行分析计算，得出当前帧的全局运动参数。

##### Step3: 运动参数滤波和图像运动补偿

图像的抖动，表现为运动参数在时间域上的高频的分量，通过滤波可以去除抖动的频率分量，从而计算出比较平滑的运动轨迹和每一帧图像要补偿的运动向量，最后通过平移和旋转等图像变换就得到无抖动的结果图像。

#### 5.3.3 运动参数估计

用绝对差和 (SAD) 对当前帧的  $16 \times 16$  的块在前一帧搜索匹配块，搜索范围为  $[-15, 15] \times [-15, 15]$ 。

上述块匹配算法找出了当前帧和前一帧的对应点集  $\{((x_i, y_i)^T; (x_i', y_i')^T)\}$ ， $i = 1, 2, \dots, n$ 。假设每帧图像的整体运动有伸缩，旋转和平移，那么图像全局运动方程可以写为：

$$\begin{bmatrix} x_i \\ y_i \end{bmatrix} = s \times \begin{bmatrix} \cos \alpha & -\sin \alpha \\ \sin \alpha & \cos \alpha \end{bmatrix} \begin{bmatrix} x_i' \\ y_i' \end{bmatrix} + \begin{bmatrix} dx \\ dy \end{bmatrix}, i = 1, 2, \dots, n \quad (5.27)$$

其中， $s$  表示图像的伸缩， $\alpha$  表示图像绕中心逆时针旋转角度， $\begin{bmatrix} dx \\ dy \end{bmatrix}$  表示图像中心的位置。式 (5.27) 还可以写成：

$$\begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} a & -b \\ b & a \end{bmatrix} \begin{bmatrix} x_i' \\ y_i' \end{bmatrix} + \begin{bmatrix} dx \\ dy \end{bmatrix} \quad (5.28)$$

其中， $s = \sqrt{a^2 + b^2}$ ， $\alpha = \tan^{-1}(b/a)$ 。

将式 (5.28) 化简可得：

$$\begin{bmatrix} x_i' & -y_i' & 1 & 0 \\ y_i' & x_i' & 0 & 1 \end{bmatrix} \times \begin{bmatrix} a \\ b \\ dx \\ dy \end{bmatrix} = \begin{bmatrix} x_i \\ y_i \end{bmatrix}, i=1,2,\dots,n \quad (5.29)$$

在上式中,  $x_i, y_i, x_i', y_i'$  已知,  $a, b, dx, dy$  未知, 两对对应点就可求解, 一般可以找到多于两对对应点, 从而 (5.29) 是一个超定方程组, 用最小二乘法可以求解。

### 5.3.4 RANSAC 算法

在图像的块匹配结果中大部分的点都有相同的运动趋势, 但也有一些点的运动向量异常, 我们称之为“坏点”。如果这些坏点数目较多或偏差较大就会使最小二乘法的结果出错。RANSAC 算法对坏点有很好的鲁棒性, 即使坏点的数目接近一半, RANSAC 算法仍然可以得出正确的结果。

RANSAC 算法和一般的最优化算法刚好相反。它对数据进行多次随机取样, 每次随机取出尽可能少但充分多个数据来确定模型参数, 再根据已确定的模型对所有数据进行划分, 一部分数据在此模型的一定误差范围内, 一部分数据在误差范围外。因为坏点是杂乱无章的异常数据, 它所确定的模型, 落在误差范围内的数据占少数, 大部分数据都落在误差范围外。而对由好点所确定的模型逼近于真实的模型, 大多数数据就会落在误差范围内。经过多次随机取样试验后, RANSAC 算法找出落在误差范围内最多的点的集合, 用此集合来做最优化, 最终确定模型的参数。

本文视频去抖动中求运动参数时采用的 RANSAC 算法描述如下:

假定块匹配的结果是在某一时刻前后两帧图像一系列对应点对的集合  $P = \{(x_i, y_i)^T; (x_i', y_i')^T\}, i=1,2,\dots,n$ , 在  $P$  中随机取两对对应点构成  $P$  的子集合  $S_1 = \{((x_i, y_i)^T; (x_i', y_i')^T), (x_j, y_j)^T; (x_j', y_j')^T\}$ , 其中  $i, j$  为两个随机数, 由  $S_1$  可以得出方程组 (5.29) 的一个解  $M_1 = (a_1, b_1, dx_1, dy_1)^T$ , 由  $M_1$  和给定的一个误差范围  $T$ , 可以得到  $P$  的子集  $S_1^*$  如下:

$$S_1^* = \left\{ ((x_i, y_i)^T; (x_i', y_i')^T) \mid \left\| \begin{bmatrix} x_i \\ y_i \end{bmatrix} - \begin{bmatrix} a & -b \\ b & a \end{bmatrix} \begin{bmatrix} x_i' \\ y_i' \end{bmatrix} + \begin{bmatrix} dx \\ dy \end{bmatrix} \right\| \leq T \right\} \quad (5.30)$$

其中, 误差范围  $T$  是一个经验值, 在系统中取  $T=5$ 。把  $S_1^*$  叫做  $S_1$  的一致集。  $S_1^*$  的元素个数为  $N_1 = \text{card}(S_1^*)$ 。

将以上过程重复  $K$  次, 并记下  $N_k$  最大的  $S_k^*$ 。用  $S_k^*$  中的元素做最小二乘法解方程组 (5.29) 得到最终结果  $M = (a, b, dx, dy)^T$ 。重复次数  $K$  由以下经验公式给出[7]:

$$K = \frac{\log(1-z)}{\log(1-w^l)} \quad (5.31)$$

式中  $w$  是在  $P$  中任取一个点落在模型误差范围内, 即满足 (5.30) 式中条件的概率, 假设  $P$  中最多有一半是坏点, 从而可以认为  $w \geq 0.5$ ,  $l$  是解方程组 (5.29) 至少所需  $P$  中元素的个数,  $l=2$ ,  $z$  是把上述过程重复  $K$  次, 至少有一次  $S_k$  中的点都是好点的概率, 也就是算法结果正确的概率, 取  $z=0.99$ 。根据经验公式可以算出  $K \approx 17$ 。也就是说, 将上述过程重复17次, 算法得出正确结果的概率为0.99。

### 5.3.5 运动参量滤波

以上求出了每帧图像相对于前一帧的运动参量  $(s_t, \alpha_t, dx_t, dy_t)^T$ , 其中  $t$  表示帧的时间

序号。在实际中,发现 $s_t$ 的值接近为1,可以近似认为它恒为1,另外还近似假设 $\alpha_t, dx_t, dy_t$ 三个分两相互独立,并且认为总体运动为这三个运动参量的叠加。若取0帧为参考帧,那么第 $t$ 帧图像相对于参考帧的运动为 $t$ 之前所有帧相对运动的叠加,也就是:

$$(\alpha_t, dx_t, dy_t)^T = (\sum_0^t \alpha_\tau, \sum_0^t dx_\tau, \sum_0^t dy_\tau)^T \quad (5.32)$$

滤除运动参量时间函数的高频分量,采用如下FIR滤波器:

$$h(t) = \begin{cases} \frac{\sin(\frac{2\pi t}{N-1})}{\frac{2\pi t}{N-1}} & -\frac{N-1}{2} \leq t \leq \frac{N-1}{2} \\ 0 & \text{其他} \end{cases} \quad (5.33)$$

它是把sinc函数截短后得到的,截至频率约为 $1/(N-1)$ 。

假设第 $t$ 帧的滤波结果为 $(\alpha_t', dx_t', dy_t')^T$ ,它代表了图像无震动的平滑移动,差值 $(\alpha_t' - \alpha_t, dx_t' - dx_t, dy_t' - dy_t)^T$ 代表了振动或偏移的运动分量。也就是要对图像要做的补偿。

### 5.3.6 图像运动补偿










知道了图像振动或偏移的运动分量,就可以对图像做运动补偿,以消除图像的抖动。对原始图像上的每一个像素点 $(x, y)^T$ ,做运动补偿后它的位置 $(x', y')^T$ 应该为:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(\alpha_t' - \alpha_t) & -\sin(\alpha_t' - \alpha_t) \\ \sin(\alpha_t' - \alpha_t) & \cos(\alpha_t' - \alpha_t) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} dx_t' - dx_t \\ dy_t' - dy_t \end{bmatrix} \quad (5.34)$$

### 5.3.7 实验结果

利用问题三附件提供的几组有晃动的视频,依次运用去抖动算法和运动目标提取算法,实验结果如表5.3所示:

表 5.3 去抖动算法和运动目标提取算法实验结果

视频名称	原图	去抖动后	前景提取
cars6			
cars7			
people1			



使用 RANSAC 算法可以有效的解决块匹配出错以及图像上局部物体运动的问题，准确的计算出图像的整体运动。在此基础上，做运动参数滤波和图像的运动补偿，消除了视频图像的抖动。结果显示算法有很强的鲁棒性，对一些比较复杂的情况都可以很好的消除抖动。

## 5.4 问题四：标记包含显著前景目标的帧号

### 5.4.1 问题分析

前三问解决了前景目标的提取方案以及视频去抖动问题，能有效的将前景目标从静态和动态背景中提取出来。在此基础上，本文利用问题二中提出的改进的 ViBe 算法将 8 组视频的前景目标提取出来，并保存为作为视频帧号标记的输入视频。其中前景目标提取视频序列为二值化序列，前景目标为白色区域，背景为黑色区域，我们利用前景目标面积双阈值将目标帧号和背景帧号标记出来。

### 5.4.2 模型的建立求解

通过上文所建立的前景提取模型，对附件 3 中的 8 组视频序列进行前景目标提取实验，得到较为理想的目标前景序列。本题要求得到每组视频中包含显著前景目标的视频帧号，由此本文采取通过计算前景目标面积的方法来标记出现目标的帧号。

由于得到的前景目标序列是二值图像，并且目标的像素为 1，显示为白色，背景的像素为 0，显示为黑色，因此可以计算像素值为 1 的个数得到目标的面积。然而对于动态背景的视频序列，得到的前景提取序列中偶尔会存在面积比较小的噪声，因此在进行前景提取前，需要对目标前景视频进行预处理来减少噪声。由于目标的连通域面积较大，可以设置一个面积阈值来消除背景中噪声。因此我们可以得到每一帧图像中像素为 1 的总数。

由于问题中需要检测显著的前景目标，因此前景目标的面积会很大，同时当非显著（面积小）目标进出背景时，散点图会有些许变化，所以我们可以设定目标面积双阈值来确定前景目标出现的帧号。综合考虑散点图的变化过程和实际场景中目标的大小，设定动态双阈值，将目标与背景的帧号区分出来。

### 5.4.3 实验结果分析

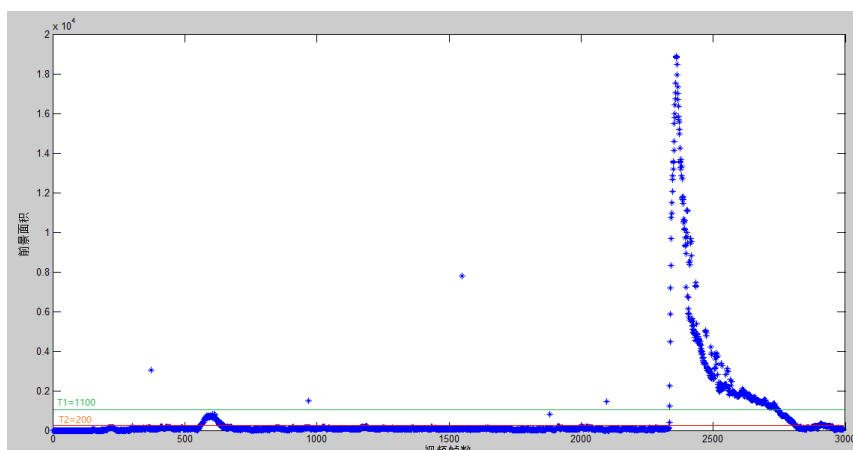


图 5.11 overpass 前景目标散点图

图 5.11 中画出了 overpass 视频序列每帧图像白色面积的散点图,可以看出视频中白色面积的变化过程(其他 7 组视频前景目标散点图见附录二)。T1 为显著目标的面积阈值, T2 为非显著目标的面积阈值, 面积大于 T1 的帧数为显著目标存在的帧数, 面积在 T1 与 T2 之间的帧数为非显著目标存在的帧数, 而面积小于 T2 的帧数为背景, 因此我们可以得到目标存在的帧数。

本文对 overpass 视频的实验结果记录在下表 5.4:

表 5.4 overpass 视频包含目标的帧号

包含显著目标的帧号				
374	968	1551	2098	2335-2799
包含非显著目标的帧号				
560-647	1881	2800-2813	2901-2919	

通过查看原始视频与得到的帧号结果表明, 视频的第 374、968、1551、2098 帧确实有显著目标出现, 而且在第 2335-2799 帧有距离摄像头较近的显著目标(人)持续运动; 在第 560-647 帧有距离摄像头较远的非显著目标(河面上的船)经过, 在第 1881 帧有船停留在河上, 在第 2800-2813、2901-2919 帧是人远去(面积较小, 属于非显著目标)的帧数。实验得出的显著目标帧号与原始视频中的帧号吻合, 并且该方法可以得到非显著目标所存在的帧号。

8 组视频包含显著前景目标的视频帧标号见表 5.5:

表 5.5 8 组视频包含显著目标的帧号

视频名称	包含显著目标的帧号
Hall	1-69、79-141、218-279、578、610-702、795、818-848、995-1065、1138、1155-1203、1246、1281-1506、1521-1543、1557-1638、1657-1707、1792-2583、3025-3248、3258-3318、3448-3540
Curtain	411、967、1349-1354、1561、1767-1900、2126、2180-2310、2642-2772-2924
Campus	85、200-224、308-445、473-488、600、645-683、692-712、742-905、1007-1035、1264、1331-1405
Lobby	79、156-199、259、347-396、521、630-660、870、965-1035、1161、1241-1286、1337-1401
Escalator	269-271、275-428、432-677、682、692-1307、1320-1321、1325-1936-1962、1967、1970-2109、2119-2390、2415、2539、2749、2754-2773-2864、2889-2979、2982、2988-3417
Overpass	374、560-647、968、1551、1881、2098、2335-2813、2901-2919
Office	197、372、501、583-2041、2080
Fountain	79-84、110-125、141、158-211、259、321-322、335、408-523

## 5.5 问题五：近似同一地点不同角度的多个监控视频中前景目标检测与提取

### 5.5.1 问题分析

上述问题一到问题四研究了单摄像机在静态背景、动态背景和摄像机发生晃动或偏移情况下的前景目标提取的算法。在视频监控系统中, 多摄像机的应用可以提供多个视角的信息。多摄像机视频监控系统中, 首先利用近似同一地点在不同高度层上的标志物, 计算基于多层的不同视角间的单应性矩阵, 然后利用混合高斯模型背景建模, 得到多个视角的前景似然图像。最后, 通过单应性变换获得目标在不同高度层的定位信息, 利用前面的运动目标提取算法进行前景目标的检测和提取。

### 5.5.2 单应性矩阵

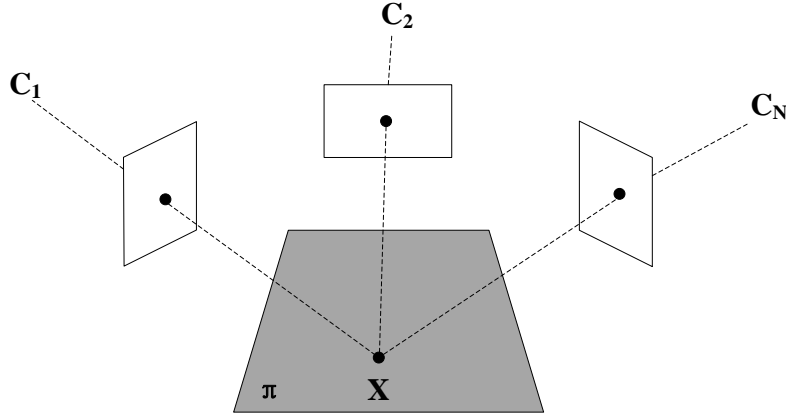


图 5.12 多摄像机示意图

如图 5.12 所示, 设  $\pi$  是不通过两个相机任一光心的空间参考平面, 本文选择地平面作为参考平面。假设在不同视角放置  $N$  个同步相机, 相机  $i$  ( $i=1, 2, \dots, N$ ) 和相机  $j$  ( $j=1, 2, \dots, N$ ) 拍摄的图像(简称像平面)分别记为  $I_i$  和  $I_j$ 。为了保证单应性的存在,  $N$  个相机必须拍摄参考平面上的同一块区域。令  $X$  是平面  $\pi$  上的任意一点,  $X$  在  $I_i$  和  $I_j$  中成像点的坐标分别  $m_k = (x_k, y_k)$  和  $m'_k = (x'_k, y'_k)$ , 我们定义一个  $3 \times 3$  的矩阵  $H_{i\pi j}$ :

$$H_{i\pi j} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \quad (5.35)$$

$H_{i\pi j}$  使  $m'_k = H_{i\pi j} m_k$ 。矩阵  $H_{i\pi j}$  称为平面  $\pi$  诱导的两个相机间(或两个像平面间)的单应性矩阵, 或简称为平面  $\pi$  的单应性矩阵, 相应的变换称为单应性变换。

利用单应性矩阵  $H_{i\pi j}$ , 从一个像平面上的点可以得到另一个像平面上的对应点。单应性矩阵  $H_{i\pi j}$  是一个齐次可逆矩阵。

### 5.5.3 基于不同高度层的目标定位

如果只在参考平面(地平面)上对目标定位, 鲁棒性比较差。当目标跑动或者跳动时, 可能和地面没有接触点, 这时在地平面上检测不到目标, 无法获得目标在地平面上的定位信息。此外, 在检测运动目标时, 无论采用哪种模型对背景进行建模, 也不能保证检测的区域完全没有缺失, 当鞋的颜色和背景颜色十分接近时, 脚的部分可能会被漏检。为了提高跟踪算法的鲁棒性, 我们在平行于参考平面不同高度的 2 个平面上也对目标进行定位, 然后融合 3 层的定位信息, 对多目标进行跟踪。选取 3 层定位信息, 既能获取目标的立体信息, 保证多目标跟踪的鲁棒性和准确性, 又可以满足实时性的要求。选取层数过多, 对提高跟踪精度没有大的帮助, 反而会导致计算效率严重下降, 无法满足实时性要求。



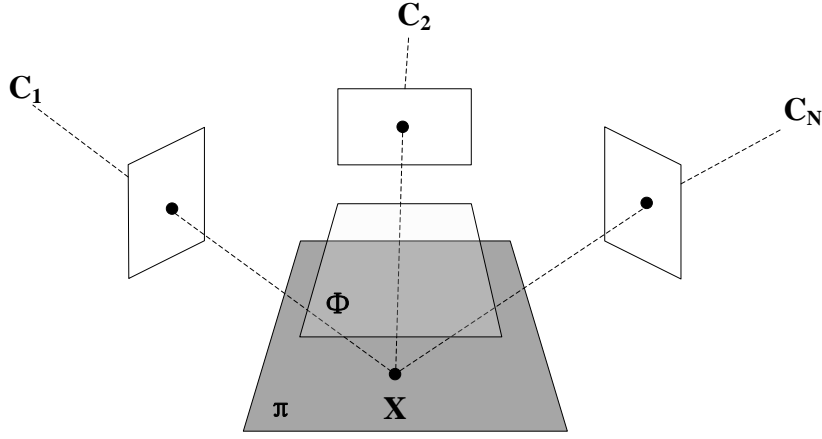


图 5.13 多层单应性示意图

如图 5.13 所示， $\pi$  为参考平面(地面)， $\Phi$  为平行于参考平面的任意平面。由平面  $\Phi$  诱导的任意两个像平面  $I_i$  和  $I_j$  间的单应性矩阵记为  $H_{i\Phi j}$ 。X 在  $I_i$  和  $I_j$  中成像点

$m_k = (x_k, y_k)$  和  $m_k' = (x_k', y_k')$ ，同样可以通过  $H_{i\Phi j} = \begin{bmatrix} h_{11}' & h_{12}' & h_{13}' \\ h_{21}' & h_{22}' & h_{23}' \\ h_{31}' & h_{32}' & 1 \end{bmatrix}$  建立映射关系，

即为：

$$\begin{bmatrix} x_k' \\ y_k' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11}' & h_{12}' & h_{13}' \\ h_{21}' & h_{22}' & h_{23}' \\ h_{31}' & h_{32}' & 1 \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ 1 \end{bmatrix} \quad (5.36)$$

同理，可以再选取一个不同高度的平面  $\kappa$ ；由平面  $\kappa$  诱导的任意两个像平面  $I_i$  和  $I_j$  间的单应性矩阵记为  $H_{i\kappa j}$ 。

利用运动目标检测算法得到每个视角的前景似然信息，选择其中任意一个视角作为参考图像，利用多层单应性映射关系将其他两个视角的前景似然信息映射到参考图像中，可以得到多个视角基于不同高度层的融合图像，融合图像上的高亮区域即为目标在不同高度层上的定位信息团块(Blob)。由此可见，基于多层的单应性矩阵的准确性十分重要，它直接关系到多层定位信息的准确性，定位信息如果不准确，那么多视角前景目标提取的准确性就无法保证了。多层前景似然图像中不可避免的会产生一些次亮点，为保证前面目标检测和提取的准确性，对每次的前景似然融合图像设立一个阈值，滤除概率较小次亮点的值，采用连通区域检测算法来分割目标在多层上的定位信息团块。

#### 5.5.4 实验结果

为了验证本文提出的基于多摄像机的目标检测与提取算法，分别在室内和室外两种光照条件下利用 3 台相机从三个视角拍摄多组实验视频进行实验，实验视频分辨率均为 320 像素×240 像素，得到三个视角的前景似然图像，我们在实验中均选取第 3 个视角作为参考图像，基于多层将其他两个视角的前景似然图像映射到参考图像上，得到基于多层的前景似然融合图像，每一层上的高亮区域，即为每个目标在每一层上的定位信息。实验结果如表 5.6 所示：

表 5.6 室外环境三个视角近似同一地点的运动目标检测和提取








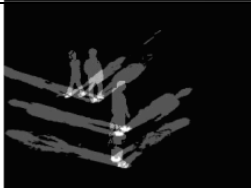




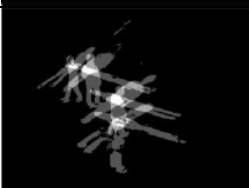
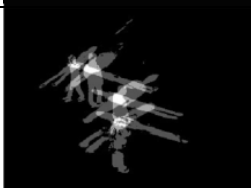
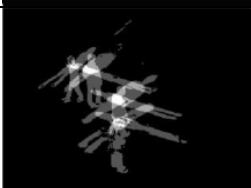















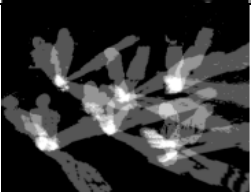
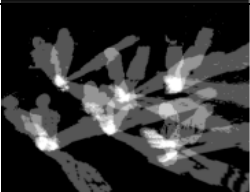
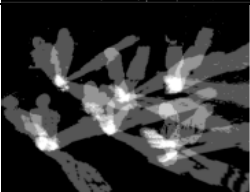
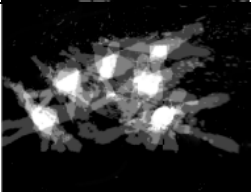
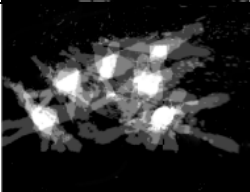
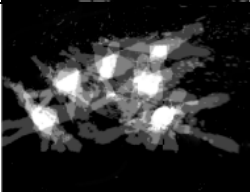
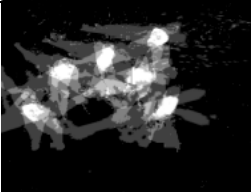
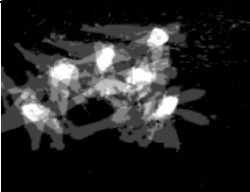
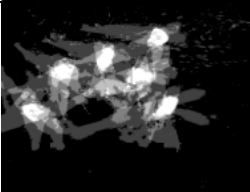









	视角 1	视角 2	视角 3
原始视频图像			
前景似然图像			
地平面融合图像			
第 2 层融合图像			
第 3 层融合图像			
地平面定位结果			
第 2 层定位结果			
第 3 层定位结果			

表 5.7 室外环境三个视角近似同一地点的运动目标检测和提取

	视角 1	视角 2	视角 3
原始视频图像			
前景似然图			
地平平面融合图像			
第 2 层融合图像			
第 3 层融合图像			
地平平面定位结果			
第 2 层定位结果			
第 3 层定位结果			

分析以上实验结果可知, 本文在运动目标检测的基础上, 利用多摄像机之间的单应性关系融合了不同视角的前景似然信息, 在不同高度的 3 个平面上对目标进行定位。这种基于多摄像机的前景目标检测与提取算法不需要标定多个摄像机, 也不需要计算多个摄像机的隐消点, 所以在很大程度上简化了基于单应性的前景目标检测和提取算法的复杂程度, 提高了单应性矩阵计算的准确性。从实验结果可以看出, 本文的算法能够满足实时性要求。

## 5.6 问题六：视频中异常事件检测

### 5.6.1 问题分析

近年来,各类重大群体性异常事件的可能性显著增加,公共安全问题日益受到国家重视。群体性异常事件(如人群短时聚集、惊恐逃散和拥挤踩踏等)从图像处理的角度来看,群体中的个体运动存在随机性和无序性,无法提取有序、规律的运动模式;场景中的复杂背景背景动态变化,而且遮挡严重,背景减除法和显著点检测等传统算法都不太适用,故很难对个体进行有效的运动检测。针对视频中异常事件的检测问题,本文采用全局异常事件(Global Abnormal Event, GAE)的方法,将异常事件检测分为两个部分:特征提取、事件建模,思路如下:

#### Step1: 光流场特征提取

提取前景图像的光流场特征。为了更加突出运动特征对异常事件检测结果的影响,本文提出光流速度分离的概念,滤除光流场中的低速部分信息,保留高速部分信息。然后使用形态学滤波消除了光流场高速部分中微弱的运动信息。

#### Step2: 事件建模

为了能够检测异常事件的发生,本文提出一种异常事件评分机制,通过计算光流场高速部分图像的均值评分。根据这个均值评分检测异常事件的发生,利用 UMN 数据库中的异常事件视频来进行实验验证异常事件检测的有效性。

本文算法的框架如图 5.14 所示:

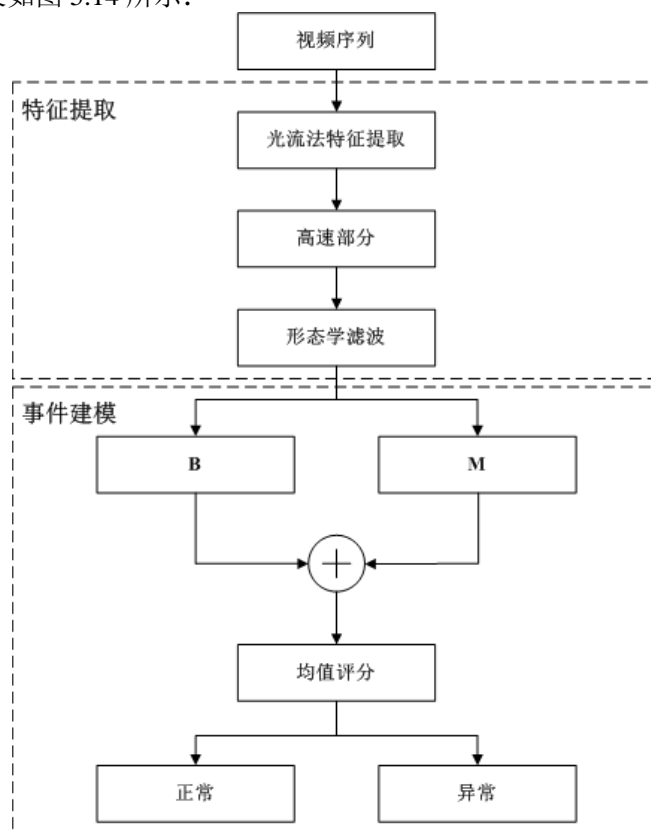


图 5.14 异常事件检测算法框架

### 5.6.2 光流场特征提取

#### (1) 光流及其估算方法

光流是一种运动模式,是目标在观察者与背景之间形成的移动。确切来说,光流是三维空间中目标速度向量在二维平面上的投影。这种在二维平面上的投影能够确切的反应出目标的运动信息,如图 5.15 所示。

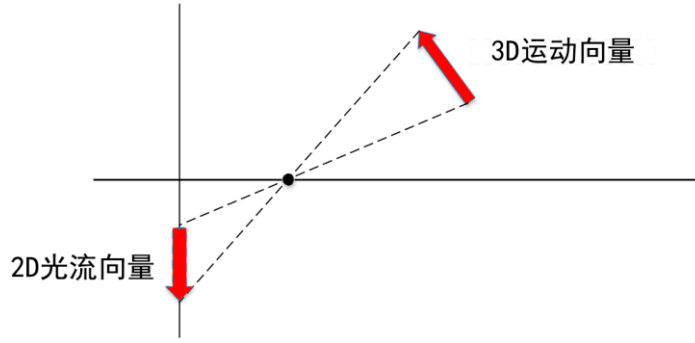


图 5.15 三维运动场在二维平面的投影图

我们令  $I(x, y, t)$  为  $t$  时刻图像  $(x, y)$  位置的光照强度， $u(x, y)$  和  $v(x, y)$  分别为该点在水平方向和竖直方向上的光流分量。那么在  $t + \Delta t$  时刻，图像  $(x, y)$  位置的光照强度可以表述为  $I(x + \Delta x, y + \Delta y, t + \Delta t)$ 。我们假设  $\Delta t$  足够小，也就是两帧之间的时间采样间隔足够小的时候，两帧之间像素点光照强度保持不变，那么我们可以得到约束方程：

$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \Delta t) \quad (5.37)$$

其中  $\Delta x = u\Delta t, \Delta y = v\Delta t$ 。所以方程也可以写成：

$$I(x, y, t) = I(x + u\Delta t, y + v\Delta t, t + \Delta t) \quad (5.38)$$

上式中有两个未知数  $u$  和  $v$ ，我们无法求解这个方程。为了求解此方程，我们需要添加一些新的假设。假设目标运动场在水平和竖直方向上的速度是连续光滑的，那么我们可以对

其进行一阶 Taylor 级数展开，可以得到：

$$I(x, y, t) \approx I(x, y, t) + \frac{\partial I}{\partial x} \Delta x + \frac{\partial I}{\partial y} \Delta y + \frac{\partial I}{\partial t} \Delta t + \varepsilon \quad (5.39)$$

此时，忽略二阶无穷小  $\varepsilon$ ，得到光流法的基本方程为：

$$\frac{\partial I}{\partial x} \Delta x + \frac{\partial I}{\partial y} \Delta y + \frac{\partial I}{\partial t} \Delta t = 0 \quad (5.40)$$

记  $u = \frac{dx}{dt}$ ， $v = \frac{dy}{dt}$ ，则公式 (5.40) 可表示为如下形式：

$$\frac{\partial I}{\partial x} u + \frac{\partial I}{\partial y} v + \frac{\partial I}{\partial t} = 0 \quad (5.41)$$

可以看出  $u$  和  $v$  分别是  $I(x, y, t)$  的光流向量中  $x, y$  的组成，而  $\frac{\partial I}{\partial x}$ ， $\frac{\partial I}{\partial y}$ ， $\frac{\partial I}{\partial t}$  是这些方向上的差分，代表光流在这些方向上的跨度。所以，我们令  $I_x = \frac{\partial I}{\partial x}$ ， $I_y = \frac{\partial I}{\partial y}$ ， $I_t = \frac{\partial I}{\partial t}$ ，则式 (5.41) 可以写成：

$$I_x u + I_y v = -I_t \quad (5.42)$$

也可以写成：

$$\nabla I \bullet \vec{V} = -I_t \quad (5.43)$$

其中  $\vec{V} = (u, v)^T$  是水平和竖直方向上的光流矢量， $I_x$ ， $I_y$  和  $I_t$  是分别沿着  $x, y, t$  三个方向上的偏导数，可以直接通过图像求解出来。由于式子 (5.43) 有两个未知量，却只有一个约束方程，所以这个方程尚不能解。要求解这个方程，我们还需要其他的约束条件。

我们假设光流  $(u, v)$  在大小为  $m * m * m (m > 1)$  的小窗格中是一个常数，那么从像素  $1, 2, \dots, n (n = m^3)$  中可以得到下列一组方程：

$$\begin{cases} I_{x_1}u + I_{y_1}v = -I_{t_1} \\ I_{x_2}u + I_{y_2}v = -I_{t_2} \\ \dots \\ I_{x_n}u + I_{y_n}v = -I_{t_n} \end{cases} \quad (5.44)$$

这个方程组只有三个未知数，但是却有多于三个的方程，也就是说方程组内有冗余。很明显这个方程组是能够求解的，方程组还可以表示为：

$$\begin{bmatrix} I_{x_1} & I_{y_1} \\ I_{x_2} & I_{y_2} \\ \cdot & \cdot \\ \cdot & \cdot \\ I_{x_n} & I_{y_n} \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} -I_{t_1} \\ -I_{t_2} \\ \cdot \\ \cdot \\ -I_{t_n} \end{bmatrix} \quad (5.45)$$

$$\text{记为:} \quad A\vec{V} = -b \quad (5.46)$$

另外，为了突出窗口中心的坐标，我们可以加上一个权重函数  $W(X)$ ，这个权重函数可以是高斯核函数或是其他核函数。那么加上权重函数后的光流场估计误差定义为：

$$E = \sum_{(x,y) \in \Omega} W^2(X)(I_x u + I_y v + I_t)^2 \quad (5.47)$$

其中  $\Omega$  是大小为  $m*m*m(m>1)$  的小窗格空间， $W(X)$  为权重函数，一般为高斯核函数， $I_x$ ， $I_y$  和  $I_t$  分别为  $x, y, t$  方向的偏导数。上式可以由最小二乘法求出结果：

$$A^T W^2 A \vec{V} = -A^T W^2 b \quad (5.48)$$

$$\text{其中,} \quad A = [\nabla I(X), \dots, \nabla I(X)]^T \quad (5.49)$$

$$W = \text{diag}[W(X_1), \dots, W(X_n)]^T \quad (5.50)$$

$$b = (I_t(X_1), \dots, I_t(X_n))^T \quad (5.51)$$

式中  $\nabla I(X) = (I_x, I_y)^T$ ，则根据以上公式，可以求得 (5.48) 的解为：

$$\vec{V} = -[A^T W^2 A]^{-1} A^T W^2 b \quad (5.52)$$

这其中  $[A^T W^2 A]$  为非奇异的时候可以得到解析解：

$$[A^T W^2 A] = \begin{bmatrix} \sum W^2(X) I_x^2(X) & \sum W^2(X) I_y(X) I_x(X) \\ \sum W^2(X) I_y(X) I_x(X) & \sum W^2(X) I_y^2(X) \end{bmatrix} \quad (5.53)$$

这其中的求和都是在大小为  $m*m*m(m>1)$  的小窗格空间  $\Omega$  的邻域得到。

## (2) 光流图像归一化

上述光流法计算得到的光流场分为水平方向和竖直方向。其光流场图像分别代表水平和竖直方向上的速度分量。但是异常行为检测的研究中，我们并不关心速度的方向，而只关心其大小。为了研究方便，我们将光流图像的两个方向进行融合，做法如公式 (5.54) 所示。

$$I = \sqrt{u^2 + v^2} \quad (5.54)$$

为了直观的显示光流场的表现，需要额外定义一些显示方式。由于一般图像的灰度级范围是  $[0, 255]$ ，我们将光流的大小归一化到这个范围内以供显示。

$$I' = \frac{I}{\text{Max}(I)} \bullet L \quad (5.55)$$

其中  $I'$  是归一化之后的图像， $I$  是待归一化图像， $\text{Max}(I)$  是图像  $I$  中的像素最大值。表示灰度级范围，这里是  $[0, 255]$ 。光流图像归一化到  $[0, 255]$  范围后，就能够在计算机上正常显示。

### (3) 光流速度分离

光流反映了视频图像的运动信息。在异常事件检测中,最容易判断异常事件发生的即是图像的运动信息。根据图像中人群的运动状态,人眼能够轻易分辨异常事件的发生:在异常事件发生的时候,通常伴随着较大的速度变化,而这种变化能够很好的通过光流场图像得到反映。通过观测异常事件视频的光流场图像我们发现,当异常事件发生的时候,由于速度的急剧增加,图像中灰色图像块逐渐变亮,并且亮块朝四周扩散。本文可以通过分析光流场亮块的变化特征判断异常事件的发生。由此本文提出光流速度分离的概念:将光流图像分为高速部分和低速部分,滤除光流场低速部分而重点分析其高速部分图像,能够更加直观的反映异常事件的发生。

光流图像速度分离的规则如公式(5.56)所示。

$$\begin{cases} I_H = I_{x,y} & I_{x,y}' > thresh \\ I_L = 0 & I_{x,y}' \leq thresh \end{cases} \quad (5.56)$$

其中,  $I_H$  是光流场高速部分,  $I_L$  是光流场低速部分,  $I_{x,y}'$  表示在图像  $(x, y)$  位置的归一化光流值, 阈值  $thresh$  是一个常数。

光流高速部分图像能够直观的反映异常事件的发生状态, 并且阈值的选取非常重要, 关系到分离后效果的好坏。如果使用固定阈值的方法, 没有办法对所有场景都适用, 所以本文采用动态阈值方法。一般的, 将阈值设为光流场图像的中值最为合适。考虑到算法运算效率的问题, 我们需要一个快速计算阈值的方法, 我们取图像最大值的  $\alpha$  倍作为阈值, 即

$$thresh = \alpha Max(I) \quad (5.57)$$

其中  $Max(I)$  是图像  $I$  中像素最大值,  $\alpha$  是一个常数, 一般取 0.25-0.5 之间。根据这一策略, 我们能够鲜明地将光流场中高速部分和低速部分进行分离。

分离过后的光流场高速部分图像中, 会有许多零散的灰白区域, 这些区域对于异常检测没有任何帮助, 反而会影响正常的检测结果。为了消除这些零散区域的影响, 对分离后的高速部分图像我们使用形态学滤波中的腐蚀操作将其滤除, 去除了许多影响异常检测结果的噪声信息。

## 5.6.3 异常事件检测

本文提出的异常事件检测算法分为三个部分: 特征提取、事件建模和事件评分。

### (1) 特征提取

提取前景图像之后, 本文使用光流场特征来描述视频图像中人群的运动信息。本文依据视频图像中人群的运动信息来判断异常事件的发生。为了凸显运动信息对异常事件检测的影响, 进一步对光流场图像进行速度分离, 滤除光流场图像中的低速部分而保留高速部分。光流图像的高速部分反应了视频中高速运动的区域, 这些区域往往是人们最感兴趣的地方, 也是判断异常事件发生与否的关键区域。但是从光流图像提取的高速部分图像中, 高速区域零散分布, 有些区域甚至只有若干微小的点, 这些微小高速区域对异常行为检测并没有任何贡献, 甚至会影响正常的异常行为检测结果。为了消除这些影响, 对高速部分图像进行形态学操作能够消除这些微小区域, 而保留面积较大块的高速区域。

### (2) 事件建模

在这一阶段, 根据光流场高速部分图像对异常事件进行建模。本文提出一种异常事件评分机制, 利用这种评分机制, 对视频图像进行评分。这种评分机制通过计算高速部分光流图像的均值评分来反映整体的运动速度变化; 通过计算高速部分光流图像的块评分来反映整体运动的聚集性。最后综合这两种评分得到异常事件总评分来有效的反映异常事件的状态变化。

### (3) 事件评分

在异常事件发生之前, 异常事件评分的分布是平缓分布的, 本文依据异常事件评分的这一分布特征提出利用高速部分光流图像的均值来判断这一变化过程, 进而保证对异常事件检测具有比较高的准确率。

### 5.6.4 异常事件评分方法

异常事件的检测需要对事件进行建模，来描述事件的状态。事件属于非结构化的数据，对事件进行合理的描述是一个值得思考的问题。本文根据视频图像中的运动信息，利用高速部分光流图像的均值评分完成对事件状态的描述。

从高速光流图像中可以看出，当人群处于比较低速运动的状态时，高速部分图像显得比较暗，图像中只有小部分区域具有不为零的像素值。而当人群处于高速运动状态的时候，高速部分光流图像显得比较明亮，大块区域像素不为零。高速部分光流图像幽暗变亮的过程能够确定异常事件的发生。根据这一特点，本文提出利用高速部分光流图像的均值来判断这一变化过程。

$$S_M = \frac{\sum I_{x,y}}{N} \quad (5.58)$$

其中  $S_M$  为图像的均值评分， $\sum I_{x,y}$  表示图像中所有像素的和， $N$  代表像素点的个数。这种均值的评分能够很好的反映出视频区域的整体运动速度，从而分辨出是否发生异常事件。

### 5.6.7 实验结果

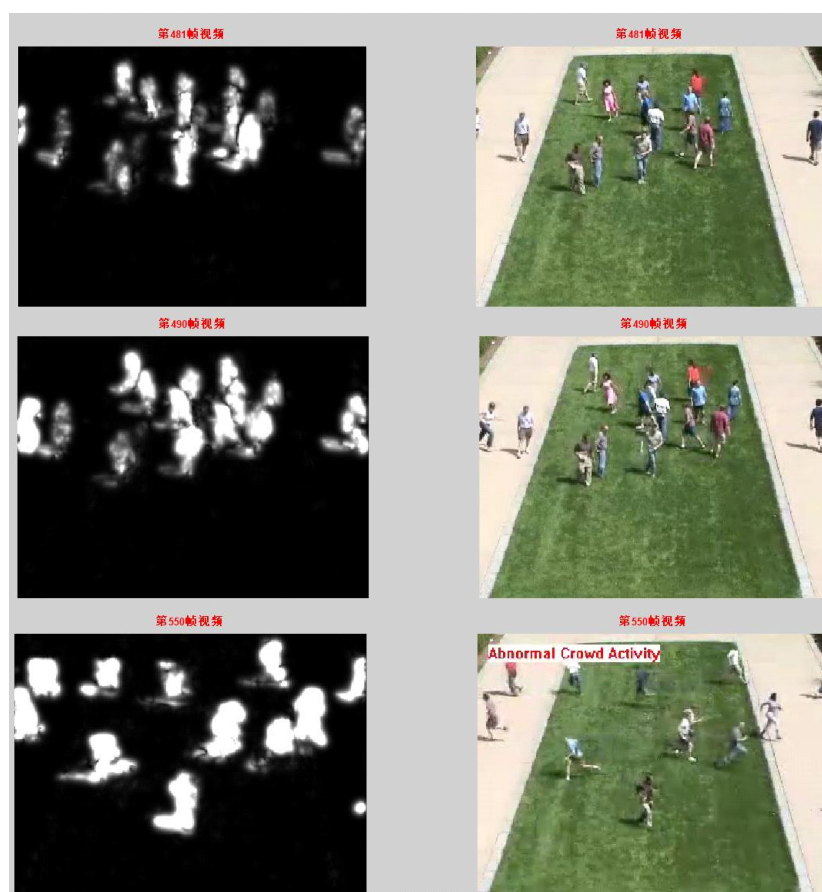


图 5.16 光流法结果图

图 5.16 中，右侧为视频原图像及所在的帧数，左侧为光流场中融合的光流结果。其中光流结果图像块越亮表示光流图像的速度越快，本文正是通过分析光流场亮块的变化特征来判断异常事件的发生。从图中可以看出，第 481 帧亮块较少，并且第 490 帧和 550 帧亮块明显增多，而实际上从原图可以看出这两帧属于人群惊慌逃散的异常事件帧。



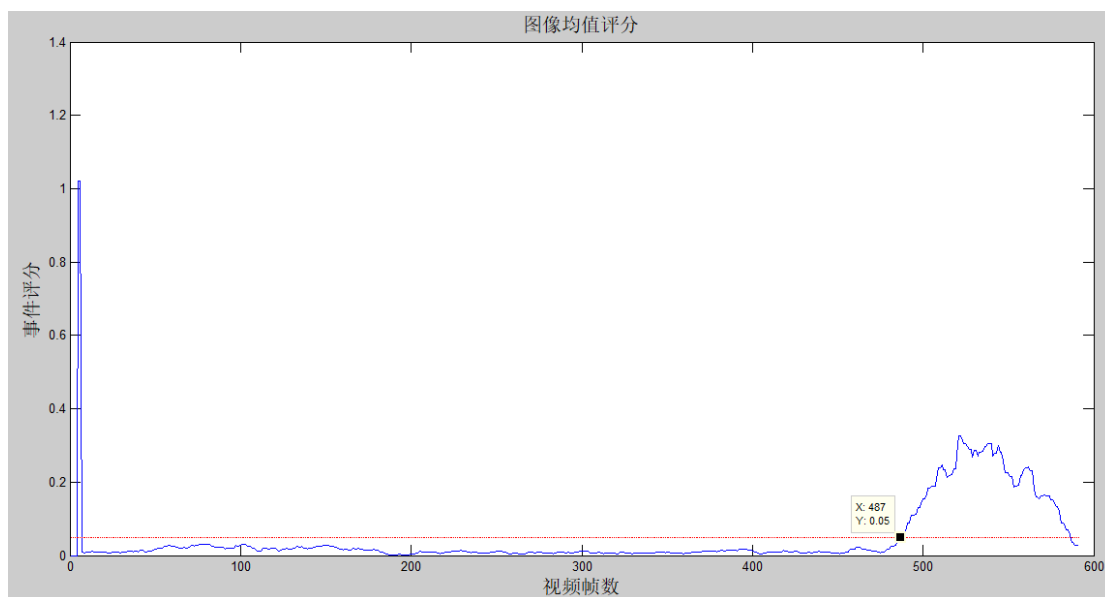


图 5.17 均值评分曲线图

如图 5.17 所示，我们通过绘制出人群惊慌逃散的图像均值评分曲线来检测所建立模型的有效性。我们设置一个阈值来区分异常事件是否发生，从图中我们可以看出，该场景一共有 593 帧画面，其中前 487 帧均值评分低于阈值，属于正常情况，第 487 帧之后开始发生异常。从实验得到的图像均值评分图中可以看出，在视频的前 487 帧之前，事件均值评分一直处于比较低和稳定的状态，在 487 帧之后评分值开始急剧增加，而且波动稍大。图中可以看出这段视频帧的变化呈现出波峰状，而实际上这个波峰状的区域就是异常事件发生的区域。由此可以看出这种评分机制模型的有效性，波峰的位置就是异常事件发生的位置。

## 6 模型的评价与改进

### 6.1 模型的评价

针对问题一、二中静态和动态背景下的前景目标提取，本文建立了多个背景模型来解决。对于本文提出的优化 ViBe 算法背景模型，能有效地提取动态背景中的前景目标，相比于传统 ViBe 算法和混合高斯模型无法解决的噪声问题，去噪能力更佳，并且该算法能适用于静态与动态背景，属于较为理想的背景模型算法。问题四提出的利用动态面积双阈值标记目标帧号的模型，能标记出视频中大约 95% 的显著目标帧号，特别对于单独帧出现的显著目标，能准确检测出其帧号，属于高准确率标记模型。问题六提出的利用目标光流特征结合均值评分机制检测异常事件的模型，能较准确地检测视频帧中异常事件发生的位置。

### 6.2 模型的改进

问题一中的传统背景建模方法得到提取效果存在噪声，后续需加强去噪能力；对于问题二中的目标提取，在光照有剧烈变化的视频帧中，优化的 ViBe 算法不能较快地适应，而且若目标长时间不移动，算法会将其判断为背景，后续需要改进的方向应将光照变化及目标长时间不移动造成的影响考虑在内；问题四中存在误标记和漏标记的帧号，这些帧大多都是受背景的影响，后续需要对分类有无目标的模型做进一步地优化；问题六中对于异常事件存在误报率，可以借鉴人工智能科学中的分类技术，选择合适的分类器快速找出样本的特点并将其正确归类，进一步完善异常事件的评分机制。

## 7 参考文献

- [1] Andrews Sobral & Antoine Vacavant, A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos, Computer Vision and Image Understanding, Volume 122, May 2014, Pages 4-21
- [2] 秦浩. 智能视频监控平台中区域目标检测和联合跟踪的设计与实现[D].南京邮电大学,2016.
- [3] 冯艳. 动态背景下基于 SIFT 特征匹配的目标检测算法[D].西安电子科技大学,2014.
- [4] Barnich O, Van Droogenbroeck M. ViBe: a powerful random technique to estimate the background in video sequences[C]//Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on. IEEE, 2009: 945-948.
- [5] 牛世伟. 基于支持向量机的多摄像机协同下运动车辆识别[D].南京理工大学,2016.
- [6] Wang H, Suter D. Background subtraction based on a robust consensus method[C]. InProc. IEEE Int. Conf. Pattern Recognit., Washington, DC, 2006: 223-226
- [7] 赵文华,姚天翔,叶秀清,顾伟康. RANSAC 算法在视频去抖动中的应用[J]. 电路与系统学报,2005,(04):91-94. [2017-09-19].
- [8] Wu Fu-Chao. Mathematical Methods in Computer Vision.Beijing: Science Press, 2008. 68-80
- [9] 章毓晋.图像工程(上册)图像处理和分析[M]. 北京清华大学出版社, 2003,259-380.
- [10] 吴新宇,郭会文,李楠楠,王欢,陈彦伦. 基于视频的人群异常事件检测综述[J]. 电子测量与仪器学报,2014,28(06):575-584.

## 8 附录

### 8.1 附录 1

#### 问题二相关程序:

##### 1.混合高斯的背景模型

```
clear all
grl = VideoReader('Campus.avi');
ft = read(grl,1);           % 把第一帧作为背景帧
ft_gy = rgb2gray(ft);
fr_size = size(ft);         %背景帧的大小
width = grl.Width;          %背景帧的宽度
height = grl.Height;        %背景帧的高度
fg = zeros(height, width);  %定义一个与背景帧灰度图像同等大小的矩阵,用于计算前景,储存前景
像素
bg_bw = zeros(height, width); %定义一个与背景帧灰度图像同等大小的矩阵,用于背景的更新
nFrames=grl.NumberOfFrames;% 帧的总数
C = 3;                      % 组成混合高斯的单高斯数目,一般取 3-5
M = 3;                      % 组成背景的数目
D = 2.5;                    % 阈值(一般 2.5 个标准差)
alpha = 0.01;               % 学习率更新速度(一般取 0.01)
thresh = 0.25;              % 前景阈值,一般为 0.25-0.75
sd_init = 6;                %初始化标准差
w = zeros(height,width,C);  %初始化权值数组
mean = zeros(height,width,C); %像素均值
sd = zeros(height,width,C);  % 像素标准差
u_diff = zeros(height,width,C); % 每个像素与均值的差
p = alpha/(1/C);            % 参数学习率
rank = zeros(1,C);          % 像素等级
%初始化均值与权值
pixel_depth = 8;             % 像素深度为 8 位
pixel_range = 2^pixel_depth -1; %像素范围 0-255
for i=1:height
    for j=1:width
```

```

        for k=1:C
            mean(i,j,k) = rand*pixel_range;    % 均值(0-255)的随机数
            w(i,j,k) = 1/C;
            sd(i,j,k) = sd_init;                % 初始标准差
        end
    end
end
% 帧处理
for n = 1:nFrames
    ft = read(ctl,n);    % 读取帧
    ft_gy = rgb2gray(ft);
    % 计算像素与均值的差值
    for m=1:C
        u_diff(:,m) = abs(double(ft_gy) - double(mean(:,m)));
    end
    % 更新每个像素的高斯
    for i=1:height
        for j=1:width
            match = 0;
            for k=1:C
                if (abs(u_diff(i,j,k)) <= D*sd(i,j,k))    % 元素匹配
                    match = 1;                            % 更新匹配
                    % 更新权重, 中值, 标准差, 参数学习率
                    w(i,j,k) = (1-alpha)*w(i,j,k) + alpha;    % 更新权重
                    p = alpha/w(i,j,k);                        % 更新参数学习率
                    mean(i,j,k) = (1-p)*mean(i,j,k) +
p*double(ft_gy(i,j));    % 更新均值
                    sd(i,j,k) = sqrt((1-p)*(sd(i,j,k)^2) + p*((double(ft_gy(i,j)) - mean(i,j,k))^2);    %
更新标准差
                else
                    % 当前像素与高斯不匹配
                    w(i,j,k) = (1-alpha)*w(i,j,k);            % 权重衰减
                end
            end
            w(i,j,:) = w(i,j,:)/sum(w(i,j,:));                % 各个像素所有高斯背景模型的值之和,
一个统计值
            bg_bw(i,j)=0;    % 用于背景的更新
            for k=1:C
                bg_bw(i,j) = bg_bw(i,j) + mean(i,j,k)*w(i,j,k); % 更新背景
            end
            % 当所有的高斯模型都不匹配的时候, 建立一个新的高斯模型, 取代权重最小的模型
            if (match == 0)
                [min_w, min_w_index] = min(w(i,j,:));        % 找出每一个像素对应的权重最小
                % 的高斯模型, 找出 w 中所有维数 (也就是所有高斯模型) 对应的第 i 行, 第 j 列的最小值 (也就是权重)
                % 返回值 min_w 为数值 (权重) 大小, min_w_index 为对应的维数 (高斯模型)
                mean(i,j,min_w_index) = double(ft_gy(i,j)); % fr_bw 为背景灰度图像, 由于没有匹配,
保留原值不变
                sd(i,j,min_w_index) = sd_init;                % 保留方差不变
            end
            rank = w(i,j,:)/sd(i,j,:);
            rank_ind = [1:1:C];                                % 计算权重与标准差的比值, 用于对
背景米线进行排序
            % 排序等级
            for k=2:C
                for m=1:(k-1)
                    if (rank(:,k) > rank(:,m))
                        % 交换最大值
                        rank_temp = rank(:,m);
                        rank(:,m) = rank(:,k);
                        rank(:,k) = rank_temp;
                        % 交换最大索引值
                        rank_ind_temp = rank_ind(m);
                        rank_ind(m) = rank_ind(k);
                        rank_ind(k) = rank_ind_temp;
                    end
                end
            end
        end
    end
end

```

```

        end
    end
    end
    %计算前景
    match = 0;
    k=1;
    fg(i,j) = 0;
    while ((match == 0)&&(k<=M))          %没有匹配，而且当前高斯数号小于背景数
        if (w(i,j,rank_ind(k)) >= thresh)
            if (abs(u_diff(i,j,rank_ind(k))) <= D*sd(i,j,rank_ind(k)))
                fg(i,j) = 0;              %判断是否是前景，如果符合匹配准则就认为是背
            end
            match = 1;
        else
            fg(i,j) = ft_gy(i,j);        %否则认为数前景，前景值为 255，即白色
        end
    end
    k = k+1;                            %计算下一个高斯模型
end
end
%      fo=imclose(fg,strel('square',12));    %腐蚀
figure(1),subplot(1,3,1),imshow(ft)        %显示输入图像
subplot(1,3,2),imshow(uint8(bg_bw))        %显示背景图像
subplot(1,3,3),imshow(fg)                  %显示前景图像
end

```

景

## 2.优化的 Vibe 算法背景模型

```

clear,clc;
video = VideoReader('input.avi');
height = video.Height;
width = video.Width;
tom = zeros(height,width);
NumFrames = video.NumberOfFrames;
cardinality = 2;%基数
r = 20;%给定半径
n = 20;%取前 20 帧作为模型初始化
%初始化
sample = zeros(height,width,n);
for nn = 1 : n
    imrgb = read(video,nn);
    imgray = rgb2gray(imrgb);
    video_dis = imgray;
    if(rem(nn,2)==0)
        sample(:,:,nn/2) = imgray;
    end
end
bg = padarray(sample,[1 1],'replicate');
%随机更新
for f = n : NumFrames
    imageRGB = read(video,f);
    imageRGB1 = read(video,f-1);
    imageRGB2 = read(video,f-2);
    imageRGB3 = read(video,f-3);
    imageRGB4 = read(video,f-4);
    imageRGB5 = read(video,f-5);
    imageRGB6 = read(video,f-6);
    imageRGB7 = read(video,f-7);
    imageRGB8 = read(video,f-8);
    image = rgb2gray(imageRGB);
    image1 = rgb2gray(imageRGB1);
    image2 = rgb2gray(imageRGB2);
    image3 = rgb2gray(imageRGB3);
    image4 = rgb2gray(imageRGB4);
    image5 = rgb2gray(imageRGB5);

```

```

image6 = rgb2gray(imageRGB6);
image7 = rgb2gray(imageRGB7);
image8 = rgb2gray(imageRGB8);
for i = 1:height
    for j = 1:width
        div = abs(sample(i,j,:) - double(image(i,j)));
        ma = find(div == (max(max(max(div)))));
        logic = div < r;
        bignum = sum(logic);
        if bignum > cardinality %确认该点为背景
            video_dis(i,j) = 0; %该点显示灰度为 0
            tom(i,j) = 0;
            %随机选择初始化背景中的一个点用新点进行替换
            randz = randi(16);
            if(randz == 10) %确定为背景点后有 1/16 的概率更新背景样本和领域样本
                rands = randi(20);
                sample(i,j,ma) = image(i,j); %要更新背景样本时从 20 个样本中选取差值最大的
            end
        end
        %随机改变(i,j)某个背景邻域像素点的某个样本值
        randy = round(rand)*2-1;%产生 0-1 之间的数，四舍五入为 0 或 1，然后乘以 2，减去 1，得到的不是-1 就是 1
        randx = round(rand)*2-1;
        sample(i+1+randy,j+1+randx,rands) = image(i,j);
    end
end
else
    if(tom(i,j) > 5) %当前像素被判定为前景点超过一定次数后，用当前像素点替换到样本集中的某个样本
        a1=abs(double(image(i,j)) - double(image1(i,j)));
        a2=abs(double(image(i,j)) - double(image2(i,j)));
        a3=abs(double(image(i,j)) - double(image3(i,j)));
        a4=abs(double(image(i,j)) - double(image4(i,j)));
        a5=abs(double(image(i,j)) - double(image5(i,j)));
        a6=abs(double(image(i,j)) - double(image6(i,j)));
        a7=abs(double(image(i,j)) - double(image7(i,j)));
        a8=abs(double(image(i,j)) - double(image8(i,j)));
        s=0;
        if(a1 < 5)
            s=s+1;
        else s=s;
        end
        if(a2 < 5)
            s=s+1;
        else s=s;
        end
        if(a3 < 5)
            s=s+1;
        else s=s;
        end
        if(a4 < 5)
            s=s+1;
        else s=s;
        end
        if(a5 < 5)
            s=s+1;
        else s=s;
        end
        if(a6 < 5)
            s=s+1;
        else s=s;
        end
        if(a7 < 5)
            s=s+1;
        else s=s;
        end
        if(a8 < 5)
            s=s+1;
        end
    end
end

```

```

else s=s;
end
if(s>6) %当前帧某像素点和前 3 帧该位置的值相差不大，说明该点是背景
    randk = randi(20);
    sample(i,j,randk) = image(i,j);
    video_dis(i,j) = 0;
else
    video_dis(i,j) = 255;
end
else
    video_dis(i,j) = 255;
    tom(i,j) = tom(i,j)+1;
end
end
end
end
randbg = randi(n);
out = bg(2:height+1,2:width+1,randbg);
video_dis= imfill(video_dis,'holes');%将原图填充孔洞
figure(1),subplot(1,2,1),imshow(image,[]);title(sprintf('第%d 帧视频', f), 'FontWeight', 'Bold', 'Color',
'r');
subplot(1,2,2),imshow(out,[]);title(sprintf('第%d 帧背景', f), 'FontWeight', 'Bold', 'Color', 'r');
subplot(1,2,2),imshow(video_dis,[]);title(sprintf('第%d 帧目标', f), 'FontWeight', 'Bold', 'Color', 'r');
drawnow;
end

```

### 问题三相关程序：

#### 1.块匹配

```

clear all
I=imread('yuantu11.jpg');%读入参考帧
J=imread('yuantu12.jpg');%读入当前帧
% A=rgb2gray(I);
% B=rgb2gray(J);
A=I;
B=J;
imshow(A);
figure ,imshow(B);
[m,n]=size(A);
dx=6;
dy=6;
block1=B(10:54,10:54); %在图像四个角的位置处选取四个块
block2=B(10:54,n-55:n-11);
block3=B(m-55:m-11,10:54);
block4=B(m-55:m-11,n-55:n-11);
for i=-dx:dx
    for j=-dy:dy
        rblock1=A(10+i:54+i,10+j:54+j);
        DB1=abs(block1-rblock1);
        SAD1(i+dx+1,j+dy+1)=sum(sum(DB1));
        rblock2=A(10+i:54+i,n-55+j:n-11+j);
        DB2=abs(block2-rblock2);
        SAD2(i+dx+1,j+dy+1)=sum(sum(DB2));
        rblock3=A(m-55+i:m-11+i,10+j:54+j);
        DB3=abs(block3-rblock3);
        SAD3(i+dx+1,j+dy+1)=sum(sum(DB3));
        rblock4=A(m-55+i:m-11+i,n-55+j:n-11+j);
        DB4=abs(block4-rblock4);
        SAD4(i+dx+1,j+dy+1)=sum(sum(DB4));
    end
end
min(min(SAD1));
[rr1,cc1]=find(SAD1==min(min(SAD1)));
r1=rr1-dx-1;
c1=cc1-dy-1;
min(min(SAD2));

```

```

[rr2,cc2]=find(SAD2==min(min(SAD2)));
r2=rr2-dx-1;
c2=cc2-dy-1;
min(min(SAD3));
[rr3,cc3]=find(SAD3==min(min(SAD3)));
r3=rr3-dx-1;
c3=cc3-dy-1;
min(min(SAD4));
[rr4,cc4]=find(SAD4==min(min(SAD4)));
r4=rr4-dx-1;
c4=cc4-dy-1;
dxx=[r1 r2 r3 r4];
dyy=[c1 c2 c3 c4];
DX=mean(r1)
DY=mean(c1)
%对当前帧进行补偿
if DX<0 %当前帧相对于参考帧向下运动
    a=zeros(abs(DX),240);
    guoduframe=[B(1+abs(DX):180,:);a];
elseif DX>0
    a=zeros(DX,240);
    guoduframe=[a;B(1:180-DX,:)];
else
guoduframe=B;
end
if DY<0; %当前帧相对于参考帧向右运动了
    c=zeros(180,abs(DY));
    buchangframe=[guoduframe(:,abs(DY)+1:240),c];
elseif DY>0
    c=zeros(180,DY);
    buchangframe=[c,guoduframe(:,1:240-DY)];
else
buchangframe=guoduframe;
end
figure,imshow(buchangframe);%补偿后的图像
C=imabsdiff(A,B);%参考帧与当前帧的差值图
D=imabsdiff(A,buchangframe);%参考帧与补偿后当前帧的差值图
figure,imshow(C)
figure,imshow(D)

```

## 2.RANSAC 算法计算帧的全局运动参数

```

%[vMask, Model] = RANSAC( mData, ModelFunc, nSampLen, ResidFunc, nIter, dThreshold )
% mData - 数据矩阵，其中每个列向量是点
% ModelFunc - 模型创建功能的句柄。 它必须创建一个来自 nSampLen 列的向量组织的模型矩阵
% nSampLen - ModelFunc 的点数
% ResidFunc - 残差计算功能的句柄。 如参数这个函数采用模型，由...计算
% ModelFunc 和数据矩阵（全部或可能的一部分）
% nIter - RANSAC 算法的迭代次数
% dThreshold - 残渣的阈值
% 返回：
% vMask - 为内联设置 1，异常值为 0
% 模型 - 此数据的近似模型
function [vMask, Model] = RANSAC( mData, ModelFunc, nSampLen, ResidFunc, nIter, dThreshold )
% 检查参数
if length(size(mData)) ~=2
    error('Data must be organized in column-vecotors massive');
end
nDataLen = size(mData, 2);
% 初始化
Model = NaN;
vMask = zeros([1 nDataLen]);
nMaxInlyerCount = -1;
% 主循环
for i = 1:nIter

```

```

% 采样
SampleMask = zeros([1 nDataLen]);
while sum( SampleMask ) ~= nSampLen
    ind = ceil(nDataLen .* rand(1, nSampLen - sum(SampleMask)));
    SampleMask(ind) = 1;
end
Sample = find( SampleMask );
% 创建模型
ModelSet = feval(ModelFunc, mData(:, Sample));
for iModel = 1:size(ModelSet, 3)
    CurModel = ModelSet(:, :, iModel);
    % 模型估计
    CurMask = ( abs( feval(ResidFunc, CurModel, mData)) < dThreshold );
    nCurInlyerCount = sum(CurMask);
    if nCurInlyerCount > nMaxInlyerCount
        nMaxInlyerCount = nCurInlyerCount;
        vMask = CurMask;
        Model = CurModel;
    end
end
end
return;

```

#### 问题四相关程序：

```

% 检测包含前景目标的帧数
clear
video = VideoReader('Campus_output.avi'); % 读取已经做完前景目标提取的视频
NumFrames = video.NumberOfFrames;
for n=1:NumFrames
    imrgb = read(video,n);
    imgray = rgb2gray(imrgb);
    imgray=bwareaopen(imgray,20,4); % 去除较小的噪声
    % 查找每帧图片中面积最大的目标区域
    [L,nn]=bwlabel(imgray,8);
    areamax=0;
    for k=1:nn
        b=find(L==k);
        area=length(b);
        if area>areamax
            areamax=area;
        end
    end
    S(n)=areamax; % 每帧图片白色区域的面积存放在 S 序列
end
% 绘制视频每帧图像白色区域面积散点图并画出阈值 T
figure(1),title('Campus 视频目标面积散点图')
plot(1:NumFrames,S,'*')
xlabel('视频帧号')
ylabel('目标区域面积')
hold on
plot(1:NumFrames,T1,'r');
hold on
plot(1:NumFrames,T2,'g');
obj_exist=find(S>T2); % 输出包含目标的帧号

```

#### 问题六相关程序：

```

% 基于光流特征的均值评分算法主函数
clear all;
clc;
avi = VideoReader('Crowd2.avi');
NumFrames = avi.NumberOfFrames;
T=0.05; % 均值评分阈值
for ii=5:NumFrames-2
    im1 = read(avi,ii);
    im2 = read(avi,ii+1);

```



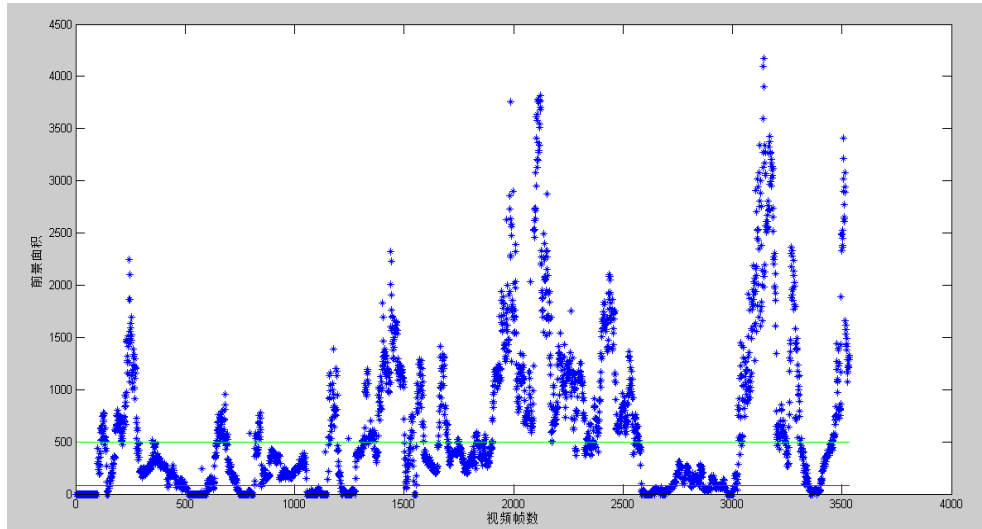
```

im3 = read(avi,ii+2);
alpha=40;
ite=30;
displayFlow=1;
displayImg=im1;
uInitial = zeros(size(im1(:,:,1)));
vInitial = zeros(size(im2(:,:,1)));
%光流场 u、v 方向的光流结果
[u, v] = HS(im1, im2, alpha, ite,uInitial,vInitial, displayFlow, displayImg);
u(isnan(u))=0;
v(isnan(v))=0;
A1 = zeros(avi.Height,avi.Width);
for i=1:avi.Height
    for j=1:avi.Width
        A1(i,j)=sqrt(u(i,j).^2+v(i,j).^2);
    end
end
[u, v] = HS(im2, im3, alpha, ite,uInitial,vInitial, displayFlow, displayImg);
u(isnan(u))=0;
v(isnan(v))=0;
A2 = zeros(avi.Height,avi.Width);
for i=1:avi.Height
    for j=1:avi.Width
        A2(i,j)=sqrt(u(i,j).^2+v(i,j).^2);
    end
end
A3 = A1+A2;
%均值评分模型
N=avi.Height*avi.Width;
A3_max=max(max(A3));
for i=1:avi.Height
    for j=1:avi.Width
        if A3(i,j)<1
            A3(i,j)=0;
        end
    end
end
S(ii)=sum(sum(A3))/N;
%显示光流结果图与原图
figure(1), subplot(1,2,1), imshow(A3), title(sprintf('第%d 帧视频', ii),
'FontWeight', 'Bold', 'Color', 'r');
subplot(1,2,2), imshow(im1), title(sprintf('第%d 帧视频', ii), 'FontWeight',
'Bold', 'Color', 'r');
end
%绘制均值评分曲线图
figure, plot(S), title('图像均值评分');
plot(1:size(S,2),T, 'r'); xlabel('视频帧数'); ylabel('事件评分');

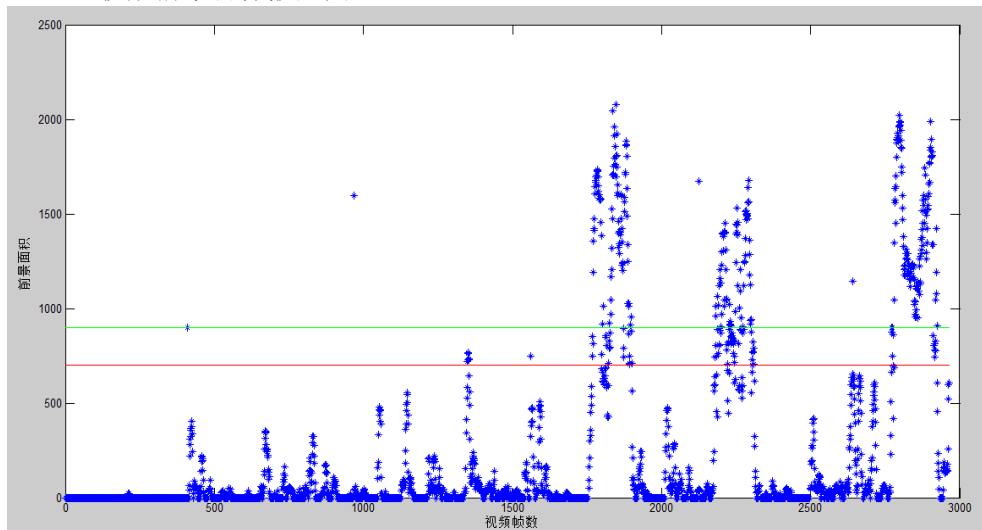
```

## 8.2 附录 2

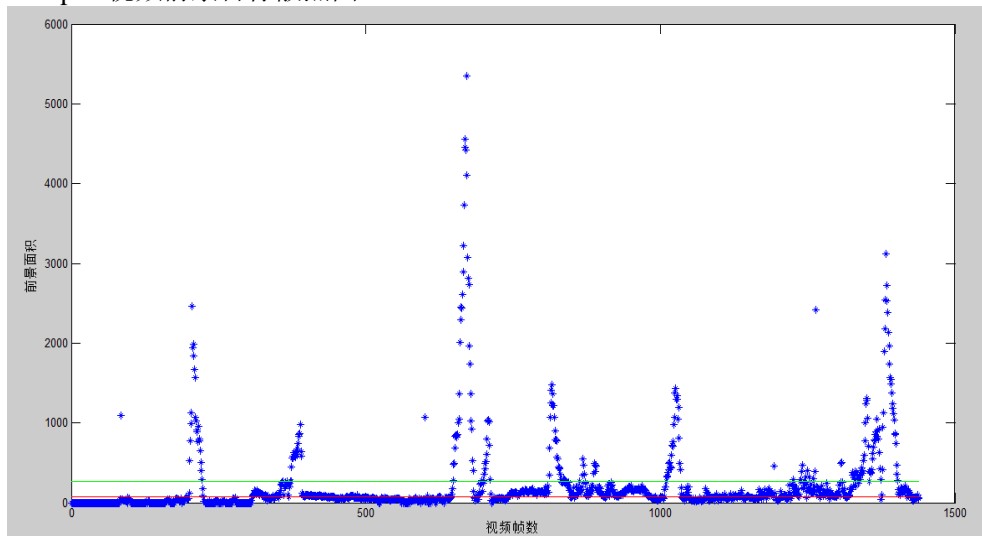
Hall 视频前景目标散点图:



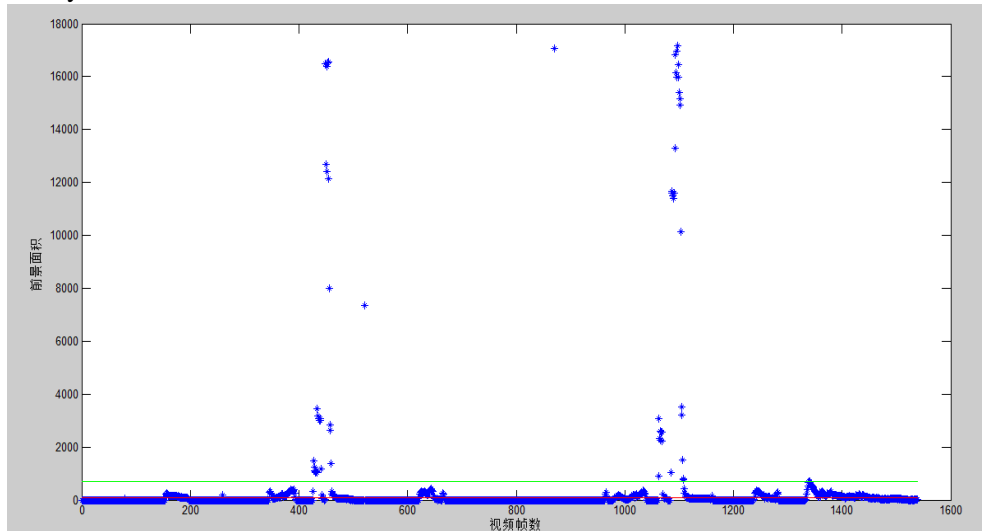
Curtain 视频前景目标散点图:



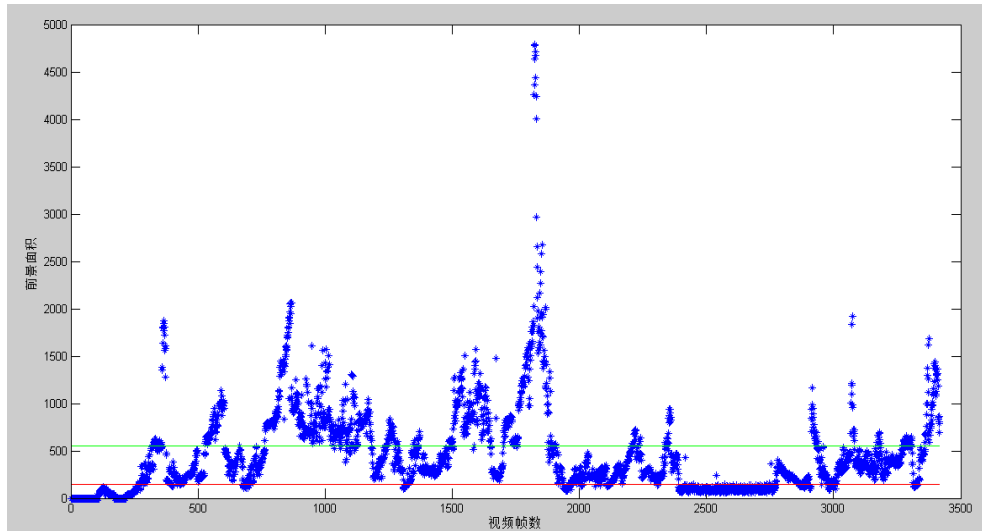
Campus 视频前景目标散点图:



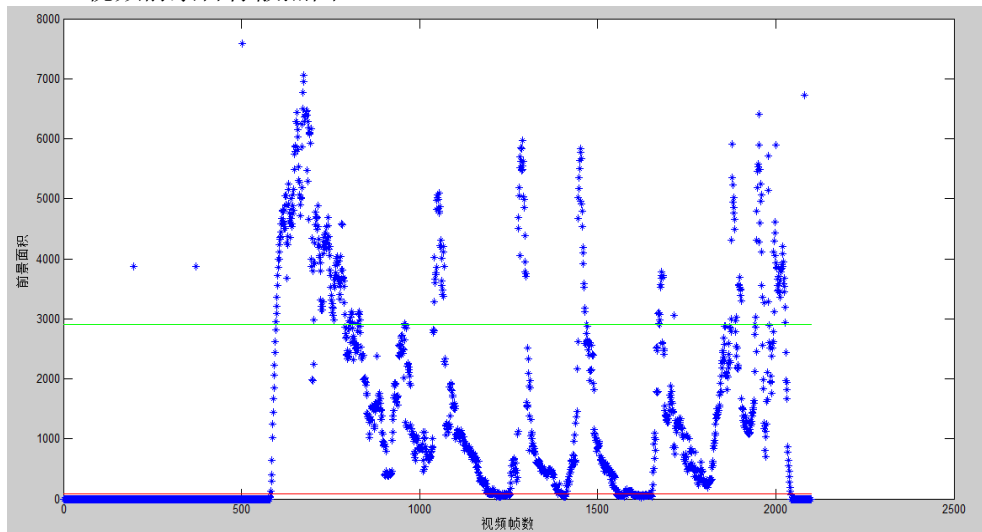
Lobby 视频前景目标散点图:



Escalator 视频前景目标散点图:



Office 视频前景目标散点图:



Fountain 视频前景目标散点图:

