华中科技大学

# 本科生毕业设计（论文）参考文献译文本

译文出处：Olivier Barnich and Marc Van Droogenbroeck.

ViBe: A Universal Background Subtraction Algorithm for Video Sequences,
IEEE TRANSACTIONS ON IMAGE PROCESSING,
NO. 6, JUNE 2011, VOL. 20:1709~1724


院　　系 人工智能与自动化学院

专业班级 自动化卓越计划实验班 1601

姓　　名 邓睿

学　　号 U201614456

指导教师 田金文


2020 年 2 月

# 译文要求

一、 译文内容须与课题（或专业内容）联系，并需在封面注明详细出处。

二、 出处格式为

图书：作者.书名.版本（第×版）.译者.出版地：出版者，出版年.起页～止页

期刊：作者.文章名称.期刊名称，年号，卷号（期号）：起页～止页

三、 译文不少于 5000 汉字（或 2 万印刷符）。

四、 翻译内容用五号宋体字编辑，采用 A4 号纸双面打印，封面与封底采用浅蓝色封面纸（卡纸）打印。要求内容明确，语句通顺。

五、 译文及其相应参考文献一起装订，顺序依次为封面、译文、文献。

六、 翻译应在第七学期完成。

# 译文评阅

**导师评语**

应根据学校“译文要求”，对学生译文翻译的准确性、翻译数量以及译文的文字表述情况等做具体的评价后，再评分。

评分：_____（百分制）　　指导教师（签名）：_____

年　　月　　日

# ViBe: 用于视频序列的通用背景消除算法


## 1 摘要

本文提出了一种结合了多种创新机制的运动检测技术。例如，我们提出的技术为每个像素存储过去在相同位置或附近获取的一组值。然后将其与当前像素值进行比较，以确定该像素是否属于背景，并通过从背景模型中随机选择要替换的值来适应模型。这种方法不同于基于经典观念的方法，即最老的值应首先被取代。最终，当发现像素是背景的一部分时，其值将传播到相邻像素的背景模型中。我们将详细介绍我们的方法（包括伪代码和使用的参数值），并将其与其他背景消除技术进行比较。效率数字表明，在计算速度和检测率方面，我们的方法均优于最新的和经过验证的最新方法。我们还将分析算法的缩减版本的性能降低到相对比较值来说在每个像素需要多少字节的内存上的绝对最小值。看起来，即使是我们算法的这种简化版本也比主流技术的性能更好。



## 2 索引词

背景消除，计算机视觉，图像运动分析，图像分割， 学习（人工智能），像素分类，实时系统，监视，视觉和场景理解，视频信号处理。


## 3 介绍

在过去十年中，全球范围内可用的相机数量急剧增加。但是，这种增长导致数据大量增加，这意味着数据无法存储或手动处理。为了自动检测，分割和跟踪视频中的对象，可以采用几种方法。简单的运动检测算法将静态背景帧与视频场景的当前帧逐像素进行比较。这是背景去除的基本原理，可以将其表述为建立背景模型并将该模型与当前帧进行比较的技术，以检测出现显着差异的区域。

因此，背景消除算法的目的是将运动对象（以下称为前景）与场景的静态或慢速运动部分（称为背景）区分开。注意当一个静态对象开始移动后，背景消除算法会检测到运动中的物体以及背景中留下的鬼影（称为“ghost”）。显然，鬼影与运动无关，必须将其丢弃。对背景的另一种定义是，它对应于大多数情况下具有可见值的参考框架，即出现概率最高，但是这种框架在实践中并不容易使用。

虽然静态背景模型可能适合在受限室内环境中分析短视频序列，但该模型在大多数实际情况下均无效。因此，需要一个更复杂的模型。此外，运动的检测通常只是理解场景的第一步。例如，可以对检测到运动的区域进行过滤和特征化，以检测无人看管的行李，步态识别，面部检测，人数统计，交通监控等。场景背景和应用的多样性解释了为什么有无

数篇论文讨论了与背景消除有关的问题。

在本文中，我们提出了一种用于背景消除的通用方法。 在[1]和专利[2]中已经简要描述了该方法。在第二部分中，我们广泛回顾了背景消除算法的文献。这篇综述介绍了为背景消除开发的主要框架，并强调了它们各自的优势。为了将它们与我们的方法进行比较，我们已经实现了其中一 些算法。第三部分介绍了我们的技术，并详细介绍了我们的主要创新：背景模型，初始化过程和更新机制。第四部分讨论了实验结果，包括与其他最新算法和计算性能的比较。我们还给出了算法的简化版本，该算法每个像素仅需要一次比较和一个字节的内存；对于任何背景消除技术而言，这都是比较和存储方面的绝对最小值。我们证明，即使是简化形式，我们的算法也比更复杂的技术表现更好。第五节总结全文。

## 4 背景消除算法综述

背景消除技术解决的问题是将观察到的图像与不包含任何感兴趣对象的估计图像进行比较；这涉及所谓的背景模型（或背景图像）[3]。这个比较过程称为前景检测，它将观察到的图像分为覆盖整个图像的两个互补像素集：1）包含感兴趣对象的前景； 2）及其互补集，背景。正如[4]所指出的，很难指定一个黄金标准的定义，即背景消除技术应该检测到什么是前景区域，因为前景对象的定义是相关于应用级别的。

已经提出了许多背景消除技术以及许多模型和细分策略，并且针对此题目进行了一些调查（例如，参见[3]－[9]）。一些算法关注于理想的背景消除技术可以或应该满足的特定要求。根据[7]，背景去除技术必须适应逐渐或快速的光照变化（一天中的时间变化，云层等），运动变化（相机振动），高频背景物体（例如树的叶子或树枝）以及背景几何形状的变化（例如停放的汽车）。某些应用需要将背景去除算法嵌入到相机中，因此计算量成为主要问题。对于室外场景的监视，抗噪声的鲁棒性和对照明变化的适应性也至关重要。

文献中描述的大多数技术都对每个像素独立运行。这些技术完全将后处理算法移交给为其结果添加某种形式的空间一致性的任务。由于扰动通常会影响单个像素，因此会导致局部错误分类。相比之下，Seiki等人描述的方法。文献[10]中的假设是这样的，即背景像素的相邻块应随时间跟随相似的变化。尽管这种假设在大多数时间都有效，尤其是对于属于同一背景对象的像素，但是对于位于多个背景对象边界的相邻像素而言，这将成为问题。尽管存在这些不便，但像素仍会聚集为块，每个$N \times N$像素块被处理为$N^2$分量向量。然后随时间收集一些样本，并将其用于训练每个块的主成分分析（PCA）模型。如果一个新

视频帧的一个块接近于使用8个相邻块的PCA投影系数进行的重构，则将其分类为背景。文献[11]中也描述了这种技术，但是它缺乏一种更新机制来随着时间的推移适应块模型。在[12]中，作者集中于PCA重建误差。虽然PCA模型也使用时间样本进行训练，但重新生成的模型占了整个图像。使用当前图像与其PCA系数的图像空间中的反投影之间的简单图像差异阈值，将单个像素分类为背景或前景。对于其他基于PCA的方法，没有描述初始化过程和更新机制。

一种类似的方法，来自训练序列的序列化图像的独立分量分析（ICA），在ICA模型的训练中[13]中进行了描述。然后计算所得的混合矢量，并将其与新图像的混合矢量进行比较，以将前景与参考背景图像分开。据说该方法对室内照明变化具有很高的鲁棒性。

在[14]中引入了基于分类器的两级机制。分类器首先确定图像块是否属于背景。然后在第二阶段对背景图片进行适当的逐块更新，根据分类结果。分类算法也是其他算法的基础，如[15]中提供的那样，其中背景模型通过人工神经网络通过自我组织来学习其运动模式。基于压缩感测框架的算法通过学习和适应背景的低维压缩表示来执行背景消除[16]。该方法的主要优点在于以下事实：压缩感测无需任何辅助图像重建即可估计对象轮廓。另一方面，前景中的对象只需要占据一小部分以便正确检测摄像机视图。

背景消除在[17]中被认为是一个稀疏错误恢复问题。这些作者假设可以将视频中的每个颜色通道独立建模为其他视频帧中相同颜色通道的线性组合。因此，他们提出的方法能够通过分别为每个颜色通道找到合适的缩放比例， 从而在不改变帧组成的总体结构的情况下，准确补偿照明源的整体变化。

背景估计在[18]中被公式化为最佳标记问题，其中背景图像的每个像素都用帧号标记，指示必须复制过去的哪种颜色。作者提出的算法产生了背景图像，该背景图像是通过复制输入帧中的区域来构造的。静态背景显示出令人印象深刻的结果，但是该方法不适用于在背景中缓慢移动的对象，因为其结果是单个静态背景框架。

[19]的作者受到基于运动的感知分组的生物学机制的启发。他们提出了适用于具有高度动态背景的场景的时空显着性算法，该算法可用于执行背景消除。他们的算法与其他最新技术的比较表明，他们的算法降低了平均错误率， 但代价是处理时间过长（每帧几秒钟），这使其不适合用于实时应用。

基于像素的背景消除技术通过不断更新其模型参数来弥补空间一致性的不足。该类别中最简单的技术是使用静态背景框架（最近在[20]中使用过），（加权）运行平均值

[21]，一阶低通滤波[22]，时间中值滤波[23]，[24]，以及使用高斯[25]-[27]对每个像素进行建模。

概率方法使用Wiener [28]或Kalman [29]滤波器预测背景帧的短期演化。在[28]中，帧级组件被添加到像素级操作。其目的是检测图像的突然变化和全局变化，并相应地调整背景框。在[30]或[31]中，在高斯建模过程中，中位数模型和高斯模型可以组合在一起，以使离群值（相对于中位数）具有比离群值更大的权重。在[32]中提出了一种从视频序列中正确初始化高斯背景模型的方法。[33]中介绍的模型是一种相当简单但仍然有效的方法。它使用三个值来表示每个背景图像中的像素：最小和最大强度值，以及训练序列的连续图像之间的最大强度差。[34]的作者对模型进行了小幅改进，并引入了阴影检测和去除技术。基于$\sum-\Delta$运动检测滤波器[35]-[37]在嵌入式处理[38]，[39]中很流行。与模数转换器一样，$\sum-\Delta$运动检测滤波器由背景图像的简单非线性递归近似组成，该近似基于比较并基于基本的增量/减量（通常是-1，0和1是唯一可能的更新值）。因此，$\sum-\Delta$运动检测滤波器非常适合许多嵌入式系统缺少浮点单元。

所有这些单峰技术都可以在受控环境中获得令人满意的结果，同时保持快速，易于实现和简单。然而，当处理在复杂的环境中捕获的视频时，需要更复杂的方法，在这些环境中会遇到运动的背景，相机移动和传感器噪声高的情况[5]。

多年来，提出了越来越复杂的像素级算法。其中，迄今为止最受欢迎的是高斯混合模型（GMM）[40]，[41]。首先在[40]中提出，该模型包括通过加权高斯混合对每个像素随时间观察到的值的分布进行建模。这种背景像素模型能够应付许多实际情况的多峰性质，当遇到重复的背景运动（例如树叶或树枝）时，会产生良好的效果。自从引入以来， 该模型已在计算机视觉社区[4]，[7]，[11]，[42]-[44]中获得了广泛的欢迎，并且随着作者不断地重新审视，它仍然引起了人们的极大兴趣。该方法并提出增强算法[45]-[50]。在[51]中，提出了一种粒子群优化方法来自动确定GMM算法的参数。[52]的作者将GMM模型与基于颜色直方图和纹理信息的基于区域的算法相结合。在他们的实验中，作者的方法优于原始的GMM算法。但是，作者的技术具有相当大的计算成本，因为他们只能使用Intel Xeon 5150处理器来处理每秒$640\times480$像素的七个帧。

GMM算法的缺点在于其强大的假设，即背景比前景更常见，并且其方差明显更低。这些都不对每个时间窗口都有效。此外，如果背景中存在高频和低频变化，则无法准确调整其敏感度，并且该模型可能会适应目标本身或错过某些高速目标的检测，如[53]中所述。而

且，模型参数的估计（尤其是方差）在现实的嘈杂环境中可能会成为问题。通常，除了在硬件实现中使用固定变量之外，别无选择。最后，应该指出的是，高斯模型的统计相关性尚需讨论，因为一些作者声称自然图像显示出非高斯统计量[54]。

为了避免为概率密度函数找到合适形状的难题，一些作者将注意力转向了非参数方法来建模背景分布。非参数核密度估计方法的优势之一[53]，[55]-[59]是由于它们依赖于在像素中观察到的像素值这一事实，因此能够规避部分微妙的参数估计步骤的能力。过去，对于每个像素，这些方法通过累积从像素最近历史中采样的一组实际值来构建背景值的直方图。然后，这些方法利用该直方图估计概率密度函数，以确定当前帧的像素值是否属于背景。非参数内核密度估计方法可以通过在像素模型中直接包含新观察到的值，从而对背景中的高频事件提供快速响应。但是，由于这些方法以先进先出的方式更新其像素模型，因此这些方法成功处理以各种速度发展的伴随事件的能力令人怀疑。这导致一些作者用两个系列的值或模型来表示背景值：短期模型和长期模型[53]，[60]。尽管在某些情况下这可能是一个方便的解决方案，但仍然存在如何确定适当的时间间隔的问题。实际上，处理两个模型增加了微调底层参数值的难度。我们的方法对采样值采用了更平滑的使用寿命策略，如第四节所述，它显着提高了整体检测质量。

在codebook算法[61]，[62]中，每个像素由codebook表示，该codebook是用于长图像序列的背景模型的压缩形式。每个codebook由codewords组成，这些codewords包含通过创新的颜色失真度量转换的颜色。在[63]中已经提出了一种结合了每个像素的空间和时间上下文的改进的codebook。据信，codebook能够在有限的内存量下长时间捕获背景运动。因此，从典型的长训练序列中学习codebook，并且在[62]中描述了codebook更新机制，一旦训练阶段结束，允许算法随着光照条件而发展。但是，应该注意的是，建议的codebook更新机制不允许创建新的codewords，如果在背景发生永久性的结构变化（例如，在城市室外新空出的停车位的情况下），则可能会出现问题。

[64]，[65]的作者没有选择特定形式的背景密度模型， 而是使用了“consensus”的概念。他们为每个像素保留给定数量的最后观察到的背景值的缓存，如果新值与像素模型中存储的大多数值匹配，则将其分类为背景。人们可能会期望，这种方法将避免与偏离任意假定的密度模型有关的问题，但由于像素模型的值是根据f中的第一值替换的。 除了更新策略，它们也容易出现前面讨论过的问题，例如背景中的慢和快运动问题，除非存储大量像素样本。作者指出，要使该方法有用，最少需要20个样本的高速缓存，但他们还注意

到，对于60个以上样本的高速缓存，没有明显的进一步改进。因此，其算法的训练周期必须至少包含20帧。最后，为了应对照明变化和出现的物体或 在后台逐渐消失的情况下，consensus算法添加了两种其他机制（一种在像素级别，另一种在blob级别）来处理整个对象。

本文提出的方法在处理新背景对象或褪色背景对象时有所不同，而无需明确考虑它们。除了更快之外，我们的方法还表现出有趣的不对称性，其中，鬼影（一旦静态对象开始移动，就会发现背景区域）比停止移动的对象更快地添加到背景模型中。本文的另一个主要贡献在于提出的更新策略。基本思想是收集过去的样本，并通过忽略样本值何时添加到模型中来更新样本值。此策略可确保像素模型的样本值具有平滑的指数衰减寿命，并允许我们的技术使用每个像素大小合理的独特模型来处理伴随各种速度变化的伴随事件。

## 5 VIBE：通用背景消除技术的描述

为了在实际应用中取得成功，背景消除技术必须至少处理三个方面的考虑：1）模型是什么，其操作如何？2）如何初始化模型？和3）随时间推移如何更新模型？这些问题的答案在本节的三个小节中给出。大多数论文描述了内在模型和更新机制。只有少数几篇论文讨论初始化问题，这在期望快速响应的情况下至关重要，例如在数码相机内部。此外，模型与更新机制之间通常缺乏一致性。例如，某些技术将像素的当前值p与模型的当前值b进行比较。在给定的公差下T，他们认为如果p和b之间的绝对差T低于是一个很好的匹配。为了适应时间，相对于p的统计方差T被调整。但是统计差异是由时间平均值估算的。因此，调整速度取决于采集帧速率和背景像素的数量。在某些情况下，这是不合适的，例如，帧速率由可用带宽确定的远程IP摄像机。

下面我们详细介绍一种称为视觉背景提取器（ViBe）的背景去除技术。为方便起见，我们在附录A中以类似C的代码形式提供了算法的完整版本。

### 1.1 像素模型和分类过程

在某种程度上，对于给定的色彩空间，无法为每个背景像素确定概率密度函数（pdf），或者至少无法确定统计参数（例如均值或方差）。请注意，对于高斯模型，由于均值和方差足以确定pdf，因此没有区别。尽管经典的背景消除方法和大多数主流技术都依赖于pdf或统计参数，即使不被忽略，也很少讨论其统计意义的问题。实际上，只要达到进行相关背景分割的目标就没有必要计算pdf。一种替代方法是考虑应该随着时间的推移增强统计显着性，而进行的一种方法是建立具有实际观察到的像素值的模型。基本假设是，从

随机角度看，因为已经观察到的值应该比尚未遇到的值具有更高的再次被观察到的可能性。

像[65]的作者一样，我们不选择pdf的特定形式，因为与假定的pdf模型的偏差无处不在。此外，对pdf的评估是一个全局过程，并且pdf的形状对异常值敏感。另外，对pdf的估计引起了关于要考虑的样本数量的明确问题。选择代表性样本数量的问题是所有估计过程所固有的。

如果我们将背景去除问题视为一个分类问题，则我们要对新像素值进行重新分类，以重新考虑其在所选颜色空间中的直接邻域，从而避免任何离群值的影响。这促使我们使用一组样本而不是使用显式像素模型来为每个背景像素建模。因此，不对背景像素的pdf进行估计，因此将像素的当前值与其样本集合中最接近的样本进行比较。与现有算法相比，尤其是与基于consensus的技术相比，这是一个重要的差异。将一个新值与背景样本进行比较，并且应接近一些样本值，而不是所有值的大部分。潜在的想法是，使用少量接近值比使用大量样本来估计背景像素的统计分布更为可靠。这有点类似于对pdf的四边形进行归一化，或者通过对其进行阈值化来仅考虑基础pdf的中央部分。另一方面，如果人们信任模型的值，那么仔细选择背景像素样本至关重要。因此，就背景而言，在仅背景像素应填充背景模型的意义上，需要保守分类。

形式上，我们用$v(x)$表示给定欧几里德色彩空间中在图像中位于$x$，并带有索引$i$的背景样本值$v_i$。每个背景像素$x$由一系列背景样本值（$N$个）：

$$M(x) = \{v_1, v_2, \cdots, v_N\} \tag{1}$$

在以前的帧中拍摄。现在，我们忽略了时间的概念；这将在后面讨论。

为了根据模型$M(x)$对像素值$v(x)$进行分类，我们通过定义以$v(x)$为中心半径$R$的一个球体$S_R(v(x))$，其与样本集中的最接近值进行比较。如果此球的设定交集与模型样本集合$M(x)$的基数，表示为#，大于或等于给定的阈值$\#_{min}$，则将像素$v(x)$分类为背景。更正式地说，我们将$\#_{min}$与下公式比较：

$$\#\{S_R(v(x)) \cap \{v_1, v_2, \cdots, v_N\}\} \tag{2}$$

根据(2)，像素值$v(x)$的分类涉及$v(x)$与模型样本之间的距离$N$的计算以及$N$与阈值欧几里得距离$R$的比较。此过程在图 1 中进行了说明。请注意，由于我们只对找到一些匹配项感兴趣，因此一旦找到$\#_{min}$匹配项，即可停止像素的分割过程。

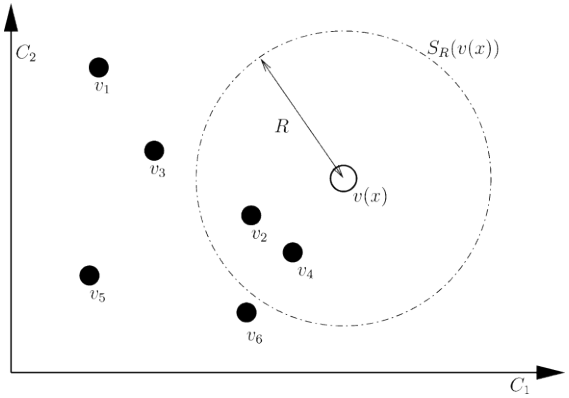


图 1 在二维欧几里得色彩空间$(C1, C2)$中像素值与一组样本的比较。为了对$v(x)$进行分类我们计算以$v(x)$为中心与半径$R$的球面相交的$M(x)$的样本数。

可以很容易地看出，我们模型的准确性仅由两个参数确定：球体的半径$R$和最小基数$\#_{min}$。实验表明，唯一的半径$R$为 20（对于单色图像）和 2 的基数是合适的。不需要在背景去除期间调整这些参数，也不需要为图像内的不同像素位置更改它们。请注意，由于样本$N$和$\#_{min}$的数量被选择为固定的，并且会影响相同的决策，因此可以使用以下比率来调整模型的灵敏度：

$$\frac{\#_{min}}{N} \tag{3}$$

但在所有比较测试中，我们保持这些值不变。到目前为止，我们已经详细说明了模型的性质。在接下来的几节中，我们将说明如何从单个模型初始化模型框架以及如何随时间更新。

1.2 单帧背景模型初始化

文献中描述的许多流行技术，例如[53]，[62]和[65]，都需要几十个帧的序列来初始化其模型。从统计的角度来看，这种方法是有意义的，因为必须收集大量数据以估计背景像素的时间分布。但是人们可能希望分割一个序列的前景，该序列比某些背景消除算法所需的典型初始化序列还要短。此外，即使存在突然的光线变化，许多应用也需要提供不间断的前景检测的能力，而算法的常规更新机制无法正确处理这种变化。这两个问题可能的解决方案是提供特定的模型更新过程，以将像素模型调整为新的照明条件。但是，使用这种专用的更新过程充其量是微妙的，因为突然的照明可能会完全改变背景的色度。

一种更方便的解决方案是提供一种将从单个帧初始化背景模型的技术。使用这种技术，对突然的照明变化的响应是很简单的：丢弃现有的背景模型，并立即初始化一个新模

型。此外，对于视频监视中的短序列或嵌入运动检测算法的设备，早在序列的第二帧就能够提供可靠的前景分割具有明显的好处。

由于单个帧中没有时间信息，因此我们使用与[66]的作者相同的假设，即相邻像素共享相似的时间分布。这证明了以下事实：我们用在每个像素的空间邻域中找到的值填充像素模型。更准确地说，我们用在第一帧中的邻域中随机抽取的值来填充它们。需要选择邻域的大小，以使其足够大以包含足够数量的不同样本，同时要记住，随着邻域的大小增加，不同位置处的值之间的统计相关性会降低。根据我们的实验，事实证明，对于每个像素$640 \times 480$像素的图像，在每个像素的 8 个连接邻域中随机选择样本是令人满意的。

正式地，我们假设$t = 0$索引第一帧，并且$N_G(x)$是像素$x$位置的空间邻域，因此:

$$M^0(x) = \{v^0(y|y \in N_G(x)\} \tag{4}$$

位置$y$是遵循统一的方法随机选择的。请注意，给定的$v^0(y)$可能会被选择多次（例如，如果邻域的大小小于的$M^0(x)$基数），或者根本不会被选择。但是，如果人们承认邻域中的值是极好的样本候选者，那么这不是问题。

该策略已被证明是成功的。唯一的缺点是，第一帧的移动物体的存在将会导致一种被普遍称为鬼影的现象。根据[24]，鬼影是“被判断为移动中但并不属于任何移动的物的连接点的集合”。在这种特殊情况下，鬼影是由来自移动对象的样本的像素模型的不幸初始化引起的。在随后的帧中，对象移动并不再遮挡真实的背景，这将通过常规的模型更新过程逐步学习，使鬼影随着时间的推移而消退。幸运的是，如下午所示，我们的更新过程既确保了在有鬼影的情况下快速的模型恢复，也确保了在背景模型中缓慢地将真实的移动对象结合（incorporation）起来。

1.3 随着时间的推移更新背景模型

在本节中，我们将描述如何用每个新帧连续更新背景模型。这是一个关键的步骤，如果我们想取得准确的结果，随着时间的推移：更新过程必须能够适应灯光变化，处理出现在场景中的新物体。

5.1.1 关于更新机制的一般讨论：

我们算法的分类步骤将当前像素值$v^t(x)$直接与时间上的前一帧背景模型$M^{t-1}(x)$中包含的样本进行比较。因此，模型存储哪些样本以及持续多长时间的问题至关重要。人们可以将模型视为背景记忆或背景历史，这在文献中经常提到。更新背景历史记录的经典方法是在一定数量的帧之后或给定的时间段（通常大约几秒钟）后丢弃并替换旧值；最旧的值

将替换为新的值。尽管有其背后的原理，但这种替换原理并不是很明显，因为如果与背景值相对应，则没有理由删除有效值。

在模型中包括或不包括前景像素值的问题对于基于样本的背景消除方法总是会提出。否则该模型将无法适应不断变化的条件。归结为保守更新方案和盲目更新方案之间的选择。请注意，基于内核的 pdf 估计技术具有较柔性的更新方法。他们可以通过在包含新值之前赋予其权重来使新值的外观变得平滑。

保守的更新策略永远不会包含属于背景模型中前景区域的样本。实际上，只有将像素样本分类为背景样本，才能将其包括在背景模型中。乍一看，这样的策略似乎是显而易见的选择。实际上，由于运动对象与背景不具有相似的颜色，因此可以保证对运动对象的清晰检测。不幸的是，它还会导致死锁情况和持久的鬼影：错误地归类为前景的背景样本阻止了其背景像素模型的更新。这可以无限期地防碍背景像素模型被更新，并可能导致永久性错误分类。不幸的是，许多实际情况导致了这种情况。例如在后台模型中，除非有专用的更新机制来处理这种情况，否则以前停放的汽车空出的位置无法通过纯保守的更新方案包含。

盲（Blind）更新对死锁不敏感：无论样本是否被归类为背景，样本都会添加到背景模型中。该方法的主要缺点是对慢速移动目标的检测不佳，这些目标逐渐包含在背景模型中。一种可能的解决方案包括使用大尺寸的像素模型，该模型可以覆盖较长的时间窗口。但这是以增加的内存使用量和更高的计算成本为代价的。此外，对于先进模型更新策略，例如[53]或[65]中采用的策略，300 个样本仅覆盖 10 s 的时间窗口[每秒 30 帧(FPS)]。背景模型中仍会包含被缓慢移动的对象覆盖超过 10 秒的像素。

严格来说，当背景被遮盖时，时间信息不可用。但是背景消除是一个时空过程。为了提高技术含量，我们可以假设，正如我们在上文中所建议的那样，期望相邻像素具有相似的时间分布。因此，根据此假设，最佳策略是采用保守的更新方案并利用空间信息，以便将有关背景演变的信息注入到前景局部遮盖的背景像素模型中。此过程在修补中很常见，在该修补中，将对象移除并在附近进行取值以填充孔洞（hole）[67]。在第下文中，我们提供了一种简单有效的方法来利用空间信息，这使我们能够克服纯保守更新方案的大多数缺点。

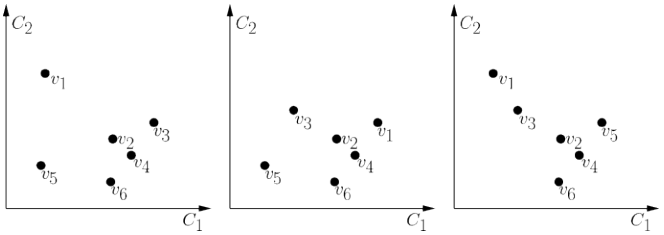


图 2 更新大小为 N = 6 的像素模型的六个可能结果中的三个。我们假设值占据的颜色空间与图 1 相同，并且我们决定更新模型。此图显示了更新后的三种可能的模型。选择一个特定模型的决策过程是随机的（概率相等）。

我们的更新方法包含三个重要组件：

1)一种无记忆更新策略，可确保存储在背景像素模型中的样本的生命周期平滑地衰减； 2)随机时间子采样（subsample）以扩展背景像素模型所覆盖的时间窗口，以及 3)机制可以在空间上传播背景像素样本，以确保空间一致性并允许对前景所掩盖的背景像素模型进行调整。在以下三个小节中，将对这些组件以及我们使用它们的原因进行描述。

5.1.2 无记忆更新策略：

许多基于样本的方法都使用先进先出策略来更新其模型。为了适当地处理场景背景中的各种事件，Wang 等人，[65]提出在像素模型中包含大量样本。但是，如前所述，这对于高帧率可能仍不足够。其他作者[53]，[58]结合了两个时间子模型来处理快速和慢速修改。这种方法被证明是有效的。然而，这增加了参数化问题，因为有必要确定更多数量的参数值以便实现实际的要求。

从理论上讲，我们认为确保样本值保留在样本集中的概率单调衰减是更合适的。像素模型应包含该像素最近的采样，但旧采样不必丢弃。

我们提出了一种为样本的剩余存在周期提供指数单调衰减的方法。该方法通过允许一些旧样本保留在像素模型中来改善估计的时间相关性。请记住，此方法与保守的更新策略结合使用，因此前景值永远不应包含在模型中。

如图 2 所示，该技术简单但有效：我们没有从像素模型中系统地删除最旧的样本，而是根据统一的概率密度函数选择随机丢弃的样本。

新值然后替换所选样本。这种随机策略与先替换旧值的想法相矛盾，这对于保守的更新策略来说是不正确的。保守的更新策略对于过程的稳定性也是必要的。事实上，随机更新策略产生了一种非确定性的背景消除算法(据我们所知，这是第一个具有该属性的背景消除算法)。只有保守的更新策略才能确保模型不会随着时间的推移而发生分歧。尽管如此，

在我们的背景消除算法同一序列不同的时间的结果之间，还是有一些细微的差别，这在我们的经验中是无法察觉的。

从数学上讲，更新像素模型后保留$t$时的样本的概率为：$(N-1)/(N)$，我们可以在以后的任何时间$t+dt$推导相似的概率，以下记为 $P(t,t+dt)$。该可能性等于：

$$P(t,t+dt)=(\frac{N-1}{N})^{(t+dt)-t} \tag{5}$$

可以重写为

$$P(t,t+dt)=e^{-\ln(\frac{N}{N-1})dt} \tag{6}$$

该表达式表明，该模型的任何样本值的预期剩余寿命都呈指数衰减。假设在时刻$t$之前将样本包含在模型中，则该样本在该时间间隔$(t,t+dt)$中保留的概率似乎与$t$无关。换句话说，过去对未来没有影响。已知该特性称为无记忆特性，适用于指数密度（参见[68]）。就我们所知，这是背景消除领域中的一项杰出且独特的属性。它使我们完全可以自由定义一个将样本保留在像素历史中的时间段，并且在某种程度上允许更新机制适应任意帧速率。

5.1.3 时间二次采样：

我们已经展示了随机替换策略如何使我们的像素模型能够以有限数量的采样覆盖大的（理论上是无限的）时间窗口。为了进一步扩展固定大小的像素模型所覆盖的时间窗口的大小，我们求助于随机时间二次采样。这个想法是，在许多实际情况下，不必为每个新帧更新每个背景像素模型。通过减少背景更新的频率，我们人为地延长了背景样本的预期寿命。但是，在存在周期性或伪周期性背景运动的情况下，使用固定的二次采样间隔可能会阻止背景模型正确地适应这些运动。这促使我们使用随机子采样策略。实际上，当像素值已被分类为属于背景时，随机过程将确定该值是否用于更新相应的像素模 型。

在所有测试中，我们采用的时间采样因子$\phi$为 16：背景像素值在 16 时有机会被选择来更新其像素模型。但有人可能希望对此进行调整参数来调整像素模型覆盖的时间窗口的长度。

5.1.4 通过背景样本传播实现空间一致性：

由于我们使用保守的更新方案，因此必须提供一种更新前景隐藏的背景像素模型的方法。一种流行的方法是使用$W^4$算法[33]的作者所称的“检测支持端口图”，该功能对像素被分类为前景的连续次数进行计数。如果此数字达到特定像素位置的给定阈值，则将该位

置的当前像素值插入背景模型中。一个变体包括在背景中包含已连接很长一段时间的静态前景像素连接组，如[69]中所述。一些作者，例如 $W^4$算法的作者和 SACON 模型的作者[64]，[65]使用像素级和对象级背景更新的组合。

使用保守更新的优势来自以下事实：被分类为前景的像素值永远不会包含在任何背景像素模型中。虽然方便，但与支持图（support map）相关的方法仅延迟了前景像素的包含。此外，由于这些方法依赖于双值决策（binary decision），因此需要花费时间才能从背景模型中对真实前景对象不正确地包含里恢复。在背景像素模型中逐步包含前景样本更为合适。

如前所述，我们有不同的方法。我们认为相邻的背景像素共享相似的时间分布，并且像素的新背景样本也应更新相邻像素的模型。根据此策略，将不时用来自相邻像素位置的背景样本更新前景隐藏的背景模型。这允许有关背景演变的信息在空间上扩散，该信息依赖于仅归类为背景的样本。因此，我们的背景模型能够依靠严格的保守更新方案来适应变化的光照和结构演变（添加或移除的背景对象）。

更准确地说，让我们考虑像素$x$的 4 或 8 连接的空间邻域，即$N_G(x)$ ，并假定已决定通过插入$v(x)$来更新样本集$M(x)$。然后，我们还使用此值$v(x)$更新样本集$M(y \in N_G\ (x))$从邻域中选择一个像素，并根据统一方法随机选择。

由于像素模型包含许多样本，因此可能会意外地插入邻域模型中的无关信息不会影响检测的准确性。此外，不相关信息的错误传播由于需要在其进一步传播之前匹配观察值而受到阻止。这种自然的局限性抑制了误差的扩散。

注意，选择策略和空间传播方法都不是确定性的。如前所述，如果算法再次在相同的视频序列上运行，结果将始终略有不同（请参见图 2）。尽管不寻常，但是允许随机过程确定要丢弃哪些样本的策略非常有效。这与引入衰落因子或使用长期和短期历史值的已知策略不同。

到此结束了对我们算法的描述。ViBe 不对视频流的帧速率或色彩空间，场景内容，背景本身或其随时间的变化做出任何假设。因此，我们将其称为通用方法。

## 6 实验结果

在本节中，我们确定 ViBe 参数的最佳值，并将其结果与其他七个算法的结果进行比较：两个简单的方法和五个最新的技术。我们还描述了 ViBe 的一些优势和内在特性，最后，我们说明了 ViBe 对于嵌入式系统的适用性。

为了进行比较，我们针对从两个测试序列中获取的帧子集生成了手动的地面真假分割图。第一个序列（称为“房屋”）在大风天室外获得。第二序列（“宠物”）来自 PETS2001 公共数据集（data-set 3, camera 2, testing）中提取的。这两个序列都具有挑战性，因为它们具有背景运动，移动的树木和灌木丛以及变化的照明条件。以下首先使用“宠物”序列来确定 ViBe 某些参数的目标值。然后，我们将 ViBe 与两个序列上的七个现有算法进行比较。

给定一系列 ground truth 分割图，许多度量可用于评估背景消除算法的输出。这些度量通常涉及以下数量：真阳性（TP），该数统计正确检测到的前景像素的数量；假阳性（FP）的数量，该误报用于计数被错误分类为前景的背景像素的数量；真阴行（TN）的数目，该数目计算正确分类的背景像素的数目；假阴性（FN）的数量，占错误分类为背景的前景像素的数量。

评估背景消除算法的困难源于缺乏标准化的评估框架。一些作者提出了一些框架，但主要目的是指出他们自己方法的优点。根据[6]，在计算机视觉中评估二分类器性能最广泛使用的度量标准是正确分类（PCC）的百分比，该百分比将所有四个值结合在一起：

$$PCC = \frac{TP+TN}{TP+TN+FP+FN} \tag{7}$$

我们的比较测试采用了该指标。请注意，为了使错误最小化， PCC 百分比需要尽可能高。

1.4 确定我们自己的参数

从前面的讨论中可以看出，ViBe 具有以下参数：

- 半径 R 用于将新像素值与像素样本进行比较的球体的范围[请参见(2)]；
- 时间二次采样因子$\phi$；
- 每个像素模型中存储的样本数量$N$；
- 和将新像素值分类为背景所需的最近像素采样数$\#_{min}$[请参见(2)]。

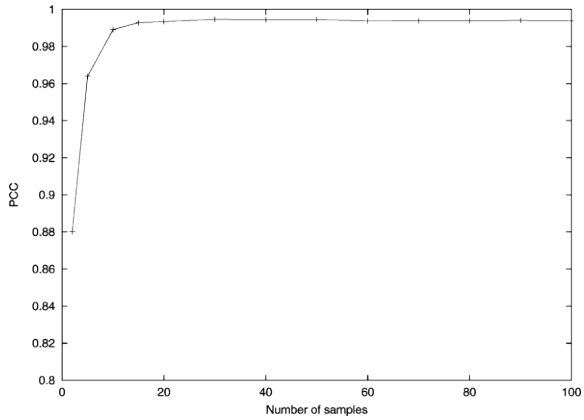


图 3 PCC 给出了在背景模型中收集的样本数量。

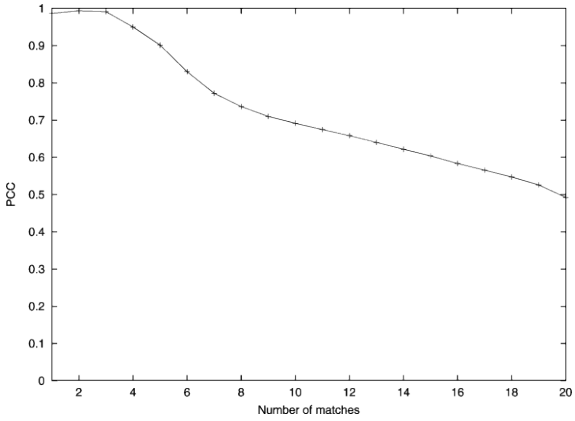


图 4 PCC 随$\#_{min}$在 1 到 20 范围变化。ViBe 的其他参数设置为$N = 20, R = 20$和$\phi = 16$。

在我们的经验中使用半径$R = 20$时间采样因子 $\phi = 16$可在各种情况下带来出色的效果。注意使用$R = 20$是一种有道理的选择，它对应于可以察觉的颜色差异。

为了确定$\#_{min}$的最佳值，我们计算了“宠物”序列上 ViBe 的 PCC 的演化，$\#_{min}$范围为 1 到 20。其他参数固定为$N = 20, R = 20$和 $\phi = 16$。**错误!未找到引用源。**显示了获得最佳 PCC 于$\#_{min} = 2$和$\#_{min} = 3$。

由于$\#_{min}$上升可能会增加 ViBe 的计算成本，因此我们将的最佳值设置为$\#_{min} = 2$。请注意，根据我们的经验，使用为$\#_{min} = 1$可以在背景稳定的场景中获得出色的效果。

为$\#_{min}$选择 2 的值后，我们将研究参数$N$的影响 ViBe 的性能。图 3 显示了从“宠物”序列获得的百分比范围为 2 到 50（$R$与$\phi$设置为 20 和 16）。我们观察到更大的$N$提供更好的性能。但是，对于高于 20 的值，它们趋于饱和,因为对于$\#_{min}$，较大的$N$值增大计算开销，我们在平台的开始选择$N = 20$。

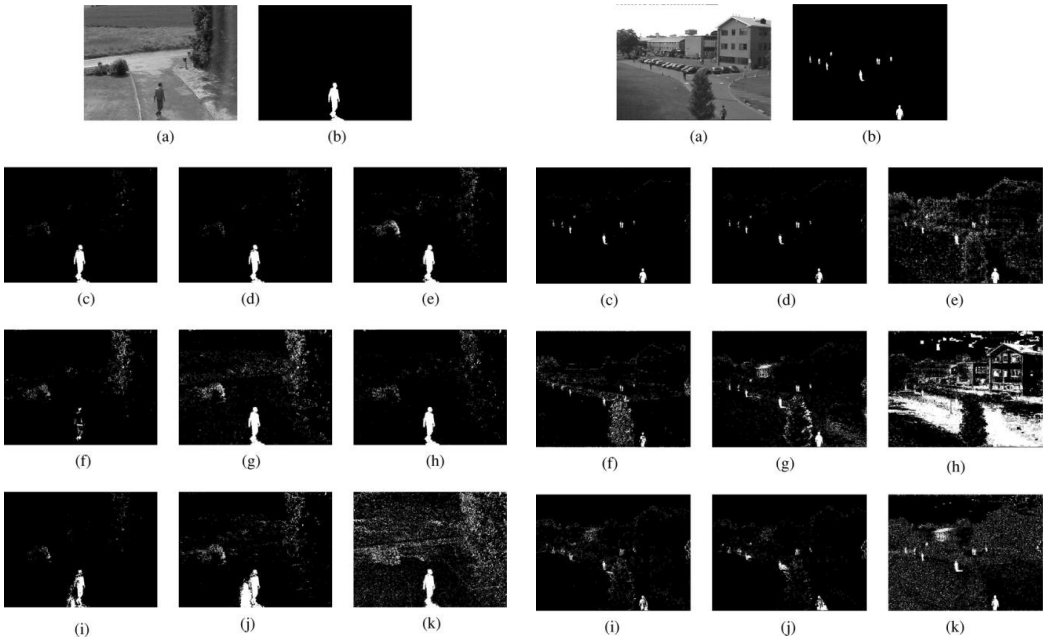


图 5 在从“房屋”序列得到的一个框架中运用九种背景消除算法得到的背景/前景分割图的对比。ViBe 的分割最接近于 ground-truth 的参考值。(a)输入图像。(b)Ground-truth 。 (c)ViBe(RGB) 。 (d)ViBe(gray)。
(e) 贝叶斯柱状图。 (f)Codebook 。 (g)EGMM[Zivokovic]。(h)GMM[Li 等]。(i)高斯模型。(j)一阶滤波器。(k)Sigma-Delta Z




图 6 从“宠物”序列中提取的一个帧的九种背景减法技术的比较背景/背景分割映射。 在这里，ViBe 的分割图也是最接近 Ground-Trut 参考。(a)输入图像。(b)Ground-truth。(c)ViBe(RGB)。(d)ViBe(gray)。
(e) 贝叶斯柱状图。 (f)Codebook 。 (g)EGMM[Zivokovic]。(h)GMM[Li 等]。(i)高斯模型。(j)一阶滤波器。(k)Sigma-Delta Z

1.5 其他技术的比较

现在，我们将 ViBe 的结果与五种最新背景消除算法和两种基本方法进行比较：1）[47]中提出的高斯混合模型（以下称为 GMM）；2）[50]的高斯混合模型（称为 EGMM）；3）基于[70]中介绍的直方图的贝叶斯算法； 4）Codebook[62]；5）[37]的 zipfian$\sum-\Delta$估计量；6）具有自适应方差的单个高斯模型（以下称为“高斯模型”）；和 7）一阶低通滤波器（即$B_t=\alpha I_t+(1-\alpha)B_{t-1}$，其中$I_t$和$B_t$分别是时间$t$的输入图像和背景图像），被用作基准。

一阶低通滤波器通过衰落系数$\alpha=0.05$和检测阈值$T=20$来测试。衰减系数$\alpha=0.05$作为衰减模型。[70]的 GMM 和[47]的贝叶斯算法使用英特尔 IPP 图像处理库中可用的实现进行了测试。对于[50]的 EGMM 算法，我们使用了作者网站上的实现。（http://staff.science.uva.nl/~zivkovic/DOWNLOAD.html）zipfian$\sum-\Delta$的作者慷慨地提供了

他们的代码供我们测试他们的方法。我们自己实现 Codebook 算法并使用如下变量：50 training frames，$\lambda = 34$，$\epsilon_1 = 0.2$，$\epsilon_2 = 50$，$\alpha = 0.4$，和 $\beta = 1.2$。

ViBe 已使用本文建议的默认值进行了测试：$N = 20$，$R = 20$，$\#_{min} = 20$, 和$\phi = 16$。大多数算法都是使用 RGB 颜色空间进行测试的；Codebook 使用自己的色彩空间，和$\sum - \Delta$滤波器实现适用于灰度图像。此外，我们实现了 ViBe 的灰度版本。

图 5 和图 6 展示出了每个序列的一个典型帧的前景检测的示例。前景和背景像素分别以白色和黑色显示。

在视觉上，ViBe 的结果看起来更好，并且最接近于真实的参考。PCC 分数证实了这一点；两种序列的九种比较算法的 PCC 得分如图 7 所示。

我们还将这九种算法的计算时间与性能分析工具进行了比较，并以可达到的帧速率表示了计算时间。图 8 显示了 它们在我们平台上的平均处理速度（2.67 GHz Core i7 CPU，6 GB RAM，C 实现）。

我们没有明确优化算法的代码，除非是它的作者对 IPP 库算法（GMM 和贝叶斯直方图）进行了优化，该算法针对 Intel 处理器和 ViBe 进行了优化。为了加快在 ViBe 中涉及随机数的操作，我们使用了预先填充了随机数的缓冲区。

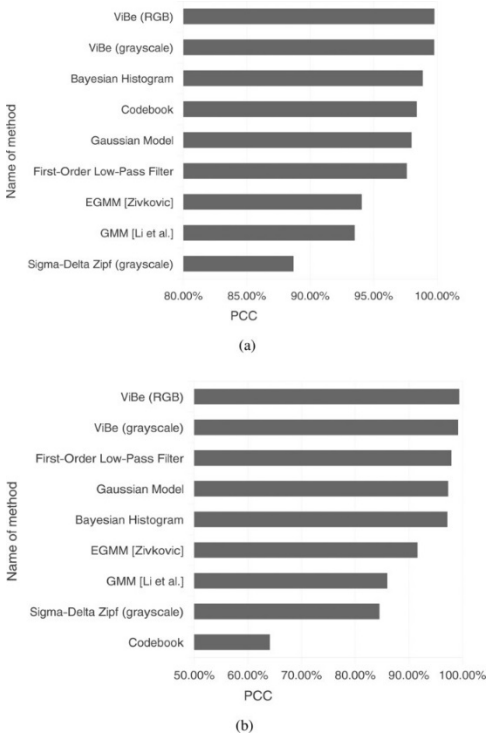


图 7 九种背景减法技术的 PCC。ViBe 具有最高的 PCC。（a）第一个序列的结果（“房屋”）。（b）第二个序列的结果（“宠物”）

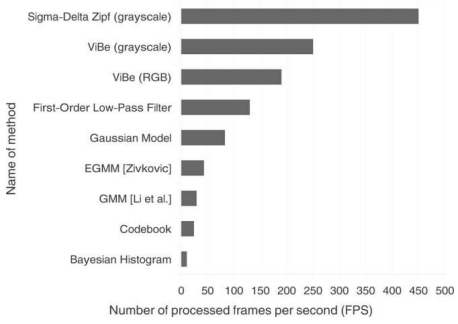


图 8 $640 \times 480$像素宽图像的九种背景减影技术的处理速度，以 FPS 表示。

我们看到 ViBe 明显优于其他七个技术：其中 PCC 在两个序列中都最高，其处理速度高达 200 FPS 的帧速率，这是英特尔优化算法的五倍。比较这些与 Chiu 等人提出的算法所获得的数据一致。[71]最近；他们声称以约 40 FPS 的帧率分割$320 \times 240$个图像。简单缩放到图像大小即可将该值降低到 10 FPS。唯一比 ViBe 快的方法是 zipfian$\sum - \Delta$估计器，其 PCC 比 ViBe 小 12 至 15%。zipfian$\sum - \Delta$算法的作者为我们提供了“房屋”序列的后处理分割图，这些分割图显示了改进的 PCC，但代价是处理速度较低。有人可能想知道，ViBe 是否有可能比一阶滤波器模型等更简单的技术运行得更快。

就 PCC 分数而言，只有基于直方图的贝叶斯算法[70]与 ViBe 竞争。但是，它比 ViBe 慢 20 倍以上。如图所示。如图 5 和图 6 所示，ViBe 的灰度和彩色版本可以将非常小的 FP 速率和对前景像素的清晰检测结合在一起。ViBe 的低 FP 率消除了对任何后处理的需求，从而进一步减轻了前景检测的总计算成本。

接下来，我们集中于 ViBe 的特定优势：快速鬼影抑制，对相机抖动的固有弹性机制，噪声弹性以及适合嵌入。

1.6 更快的鬼影抑制

背景模型必须适应因照明条件变化而引起的背景修改，也必须适应因某些部分的增加，移除或移位而引起的背景修改。这些事件是鬼影出现的主要原因：连接的前景点区域与任何真实物体都不对应。

当使用检测支持图或相关技术来检测和抑制重影时，很难（即使不是不可能）将鬼影与当前为静态的前景对象区分开。因此，如果真实前景对象保持静态时间过长，则它们将包含在背景模型中。这是正确的行为，因为静态前景对象必须在给定时间后最终成为背景的一部分。如果鬼影比真实对象更快地包含在背景模型中会更好，但这是不可能的，因为无法使用检测支持图来区分它们。

我们的空间更新机制加快了在背景模型中包含鬼影的速度，因此该过程比包含真实静态前景对象的过程要快。之所以能够做到这一点，是因为前景对象的边界经常显示出与周围背景像素模型中存储的样本明显不同的颜色。当前景物体停止移动时，前文中描述的信息传播技术会使用来自周围背景像素的样本更新位于其边界处的像素模型。但是这些样本无关紧要：它们的颜色与对象边界的所有颜色都不匹配。在随后的帧中，对象保留在前景中， 因为背景样本无法通过其边界在前景对象内部扩散。

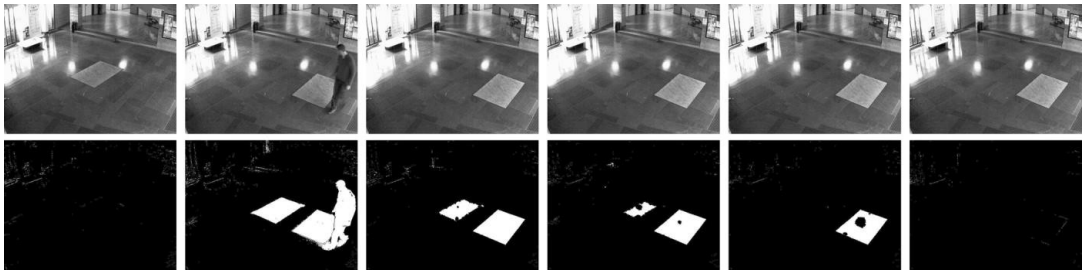

图 9 快速抑制鬼影。在此场景中，物体（地毯）被移动，在背景中留下了阴影，并被检测为前景的一部分。可以看出，鬼影被吸收到背景模型中的速度比对应于真实物理对象的前景区域快得多。

相反，鬼影区域通常与周围的背景共享相似的颜色。当来自鬼影周围区域的背景样本试图在鬼影内扩散时，它们就可能会在鬼影散布的位置与图像的实际颜色匹配。结果，鬼影逐渐被侵蚀，直到其完全消失。图 9 说明了这一讨论。

过程的速度取决于背景的纹理：在没有边缘的背景下可以获得更快的鬼影抑制。此外，如果去除的物体的颜色接近未覆盖的背景区域的颜色，则鬼影的吸收更快。必要时，可以通过调整时间二次采样因子$\phi$来调整鬼影抑制过程的速度。例如，在图 9 所示的序列中，如果我们假设帧速率为 30 FPS，则在 2 s 后重影消失，时间采样因子$\phi$等于 1。但是，如果将$\phi$设置为 64 ，ViBe 需要 2 分钟才能完全抑制鬼影。为了进行比较，贝叶斯直方图算法在$5s$内抑制了相同的鬼影区域。

人们可能会问，静态前景对象最终将如何最终包含在背景模型中。将前景像素吸收到背景中的责任在于视频序列中不可避免地会出现噪声。由于噪声，前景对象的某些像素最终出现在背景中，然后用作背景种子。因此，他们的模型被前景样本破坏了。这些样本后来由于背景样本的空间传播机制而扩散到其相邻模型中，并允许在背景中缓慢包含前景对象。

1.7 抵抗相机位移

在许多情况下，都会遇到照相机的小位移。这些小的位移通常是由于振动或风引起的，并且使用许多其他技术，它们会导致大量错误的前景检测。

我们的背景模型的空间一致性的另一个明显的好处是，针对这种小的相机运动，其鲁棒性得到了提高（见图 10）。由于样本是在相邻像素模型之间共享的，因此相机的小位移几乎不会引入错误的前景检测。

图 10 稍微移动的相机的背景/前景分割图。如果禁用了空间传播，则照相机运动会在高频区域（中央的图像）产生误报，而激活空间传播会避免很大比例的误报（右侧图像）。

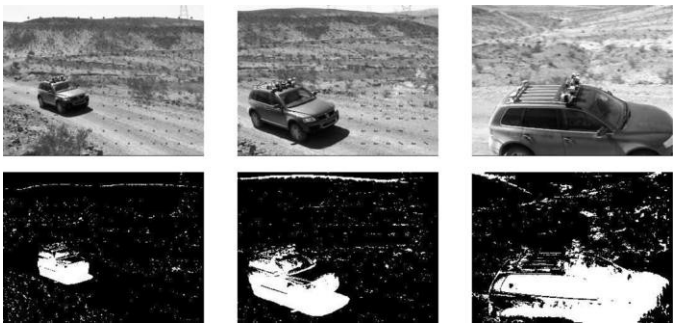

图 11 用移动相机拍摄的序列的背景/前景分割图（来自 DARPA 挑战）

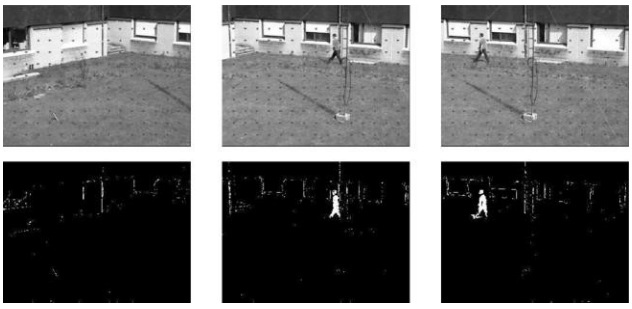

图 12 使用移动相机（监视相机）拍摄的序列的背景/前景分割图。

ViBe 还具有处理相机大排量的能力，但需要修改基本算法。由于我们的模型是完全基于像素的，因此我们可以通过允许像素模型根据相机的运动跟随相应的物理像素，使其能够处理运动的相机。可以使用嵌入式运动传感器或直接从使用算法技术的视频流。这个概念在图 11 和图 12 中示出。第一个系列显示了从 DARPA 挑战赛中拍摄的图像。摄像机从左到右平移场景，目的是跟随汽车。图 12 示出了用全倾斜变焦视频监视摄像机获取的类似场景。目的是追踪此人。

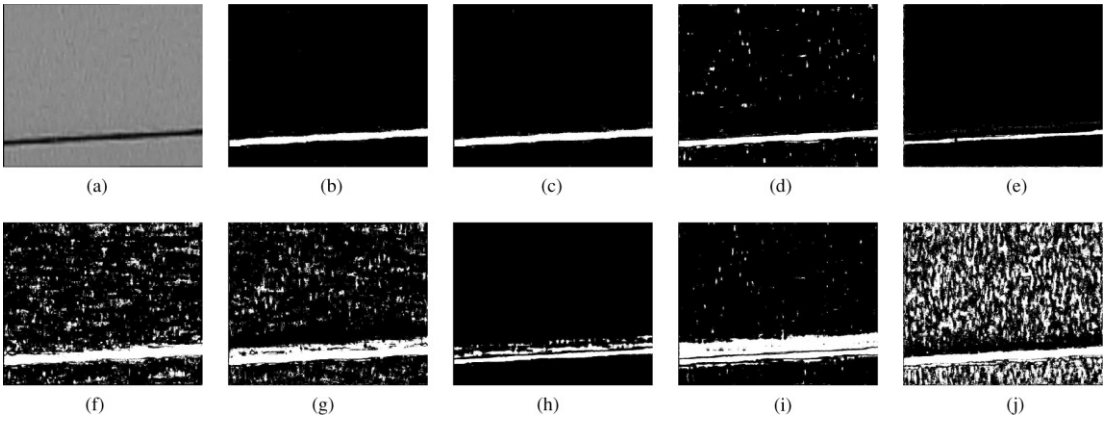


图 13 从嘈杂的“绳索”序列中获取的一帧的背景/前景分割图。（a） 输入图像。(b) ViBe (RGB). （c）ViBe（灰度）。（d）GMM［Li 等］。（e）Codebook。（f）贝叶斯直方图。（g）EGMM ［Zivkovic］。（h）高斯模型。（i）一阶过滤器。（j）$\sum - \Delta$Zipf。

为了产生图的图像。参照图 11 和图 12，使用 Lucas 和 Kanade 的光流估计器[72]，对位于规则间隔的网格上的背景点的子集，估计了两个连续帧之间的位移矢量。通过平均这些像素方向的位移矢量来计算照相机的整体位移矢量。然后根据相机在较大镶嵌参考图像内的位移重新定位像素模型。使用第前节中描述的技术瞬时初始化与第一次看到的区域相对应的像素的背景模型。可以看出，即使采用这种简单的技术，其结果也显示在图 11 和图 12。

1.8 抗噪音能力

为了证明 ViBe 对噪声的抵抗力，我们将其与其他 7 种技术在困难的噪声序列（称为“电缆”）上进行了比较。此序列显示了一个振荡的电缆，该电缆以远距光学变焦拍摄。从图 13(a)中可以看出，困难的采集条件导致像素值中出现明显的噪声。图 13 中显示的背景/前景分割图表明，ViBe 是唯一一种将低 FP 率与前景像素的精确检测结合在一起的技术。

ViBe 的高抗噪能力必须归功于两个因素。第一个源于我们的设计，允许 ViBe 的像素模型包含专门观察到的像素值。ViBe 的像素模型由嘈杂的像素值构成，因此可以自动适应噪声。第二个因素是 ViBe 使用的纯保守更新方案。通过依赖专门分类为背景的像素值，ViBe 的模型更新策略可防止在像素模型中包含任何离群值。结果，这两个因素确保了在保持相干像素模型的同时，连续适应视频序列中存在的噪声。

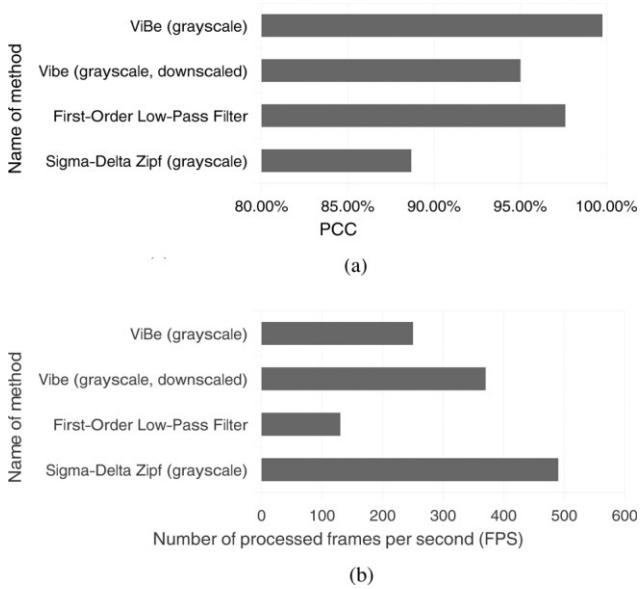


图 14 PCC 和快速技术的处理速度，以 FPS 表示，包括 ViBe 的缩小版本，该版本仅需要一个比较和每个像素一个字节的内存。(a) PCC. （b）640 × 480像素的图像的 FPS。

1.9 简化版本和嵌入式实现

由于 ViBe 具有较低的计算成本（请参见图 8），并且仅依赖于整数计算，因此特别适合嵌入式实施。此外，通过使用低价位，ViBe 的计算成本可以进一步降低和$\#_{min}$的值。图 14 中，我们给出了缩小版本的 ViBe 的 PCC 分数和帧频，该版本使用一个比较的绝对最小值和每个像素一个字节的内存。我们还提供了完整版 ViBe 的 PCC 得分和帧率, 以及前文测试中的两种更快的技术。在图 14 图的左侧，可以看到 ViBe 的缩小版本保持了较高的 PCC。请注意，它的 PCC 高于在前文中中测试的两种基于 GMM 的技术[见图 7(a)]。在处理速度或帧速率方面，[37]的 zipfian-filter$\Sigma-\Delta$方法是唯一一种比 ViBe 缩减版本更快的方法。然而，分割图的后处理步骤对于增加 zipfian $\Sigma-\Delta$方法的低 PCC 分数是必需的，并且由该后处理过程引起的计算成本显着降低了帧速率。


图 15 在佳能相机中嵌入 ViBe。

为了说明 ViBe 的低计算成本及其简单性，我们将算法嵌入到数码相机中。ViBe 在佳能 PowerShot SD870 IS 上的移植工作是使用开源替代固件 CHDK2 的修改版执行的。ViBe 的参数设置为$N = 5$和$\#_{min} = 1$。尽管相机的 ARM 处理器速度很慢，但我们还是设法平均每秒处理六帧$320 \times 240$像素宽的图像。结果如图 15 所示。

## 7 结论

在本文中，我们介绍了一种通用的基于样本的背景去除算法，称为 ViBe，它结合了三种创新技术。首先，我们提出了一个分类模型，该模型基于候选值和对应的背景像素模型之间的少量对应关系。其次，我们阐述了如何用一个帧初始化 ViBe。这使我们无需等待几秒钟即可初始化背景模型，这对于嵌入在数码相机中的图像处理解决方案以及短序列来说是一个优势。最后，我们介绍了我们的最后一项创新：独特的更新机制。与其将样本保留在像素模型中固定的时间量，我们不考虑像素在模型中的插入时间，而是选择一个要随机替换的值。这导致像素样本的平滑衰减寿命，并使该技术能够在较宽的背景演化速率范围内实现适当的性能，同时减少每个像素模型需要存储的所需样本数量。此外，我们还通过允许样本在相邻像素模型之间扩散来确保背景模型的空间一致性。我们观察到，空间过程有助于更好地适应相机运动，但它也使我们摆脱了后处理分割图以获得空间连贯结果的需要。为了有效，将空间传播技术和更新机制与严格保守的更新方案相结合：任何前景模型中均不应包含前景像素值。

在描述了我们的算法之后，我们确定了该方法所有参数的最佳值。然后使用这组参数值，将 ViBe 的分类得分和处理速度与其他两个序列的 7 种背景消除算法进行比较。事实证明，ViBe 的性能优于所有这些算法，但速度要快于其中的六个。最后，我们讨论了简化后的效果

在对于我们的算法描述后，我们决定公开所有方法中变量的最佳值。使用这系列的变量值我们将比较 ViBe 的分类得分和处理速度于其他的七种背景消除算法在两个序列上测试。ViBe 胜过所有的其他算法虽然只比其中 6 个运行的快。最后，我们讨论缩减后 ViBe 版本的规模，它可以在我们的平台上处理高于 350FPS，这种缩减版本被嵌入数码相机中去提升它对速度较慢平台的适应性，有趣的是，我们发现 ViBe 的缩减版本对于任意的背景消除算法而言都占用了相对最少的资源值（即存储的字节及每个像素相对的存储值）却就 PCC 标准来说表现得好于 state-of-the-art 算法。

它可能成为背景消除算法家庭中的一个新的里程碑。

## 8 参考文献

[1] Barnich and M. Van Droogenbroeck, “ViBe: A powerful random technique to estimate the background in video sequences,” in Proc. Int. Conf. Acoust., Speech Signal Process., Apr. 2009, pp. 945–948.

[2] M. Van Droogenbroeck and O. Barnich, Visual Background Extractor p. 36, Jan. 2009, World Intellectual Property Organization, WO 2009/007198.

[3] McIvor, “Background subtraction techniques,” in Proc. Image Vis.Comput., Auckland, New Zealand, Nov. 2000.

[4] R. Radke, S. Andra, O. Al-Kofahi, and B. Roysam, “Image change detection algorithms: A systematic survey,” IEEE Trans. Image Process.,vol. 14, pp. 294–307, Mar. 2005.

[5] Y. Benezeth, P. Jodoin, B. Emile, H. Laurent, and C. Rosenberger,“Review and evaluation of commonly-implemented background subtraction algorithms,” in Proc. IEEE Int. Conf. Pattern Recognit., Dec. 2008, pp. 1–4.

[6] S. Elhabian, K. El-Sayed, and S. Ahmed, “Moving object detection in spatial domain using background removal techniques—State-of-art,”Recent Pat. Comput. Sci., vol. 1, pp. 32–54, Jan. 2008.

[7] M. Piccardi, “Background subtraction techniques: A review,” in Proc. IEEE Int. Conf. Syst., Man Cybern., The Hague, The Netherlands, Oct. 2004, vol. 4, pp. 3099–3104.

[8] D. Parks and S. Fels, “Evaluation of background subtraction algorithms with post-processing,” in Proc. IEEE Int. Conf. Adv. Video Signal Based Surveillance, Santa Fe, New Mexico, Sep. 2008, pp.192–199.

[9] T. Bouwmans, F. El Baf, and B. Vachon, “Statistical background modeling for foreground detection: A survey,” in Handbook of Pattern Recognition and Computer Vision (Volume 4). Singapore: World Scientific, Jan. 2010, ch. 3, pp. 181–199.

[10] M. Seki, T. Wada, H. Fujiwara, and K. Sumi, “Background subtraction based on cooccurrence of image variations,” in Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit., Los Alamitos, CA, Jun. 2003, vol. 2, pp. 65–72.

[11] P. Power and J. Schoonees, “Understanding background mixture models for foreground segmentation,” in Proc. Image Vis. Comput., Auckland, New Zealand, Nov. 2002, pp. 267–271.

[12] N. Oliver, B. Rosario, and A. Pentland, “A Bayesian computer vision system for modeling human interactions,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 22, no. 8, pp. 831–843, Aug. 2000.

[13] D.-M. Tsai and S.-C. Lai, “Independent component analysis-based background subtraction for indoor surveillance,” IEEE Trans. Image Process., vol. 18, no. 1, pp. 158–167, Jan. 2009.

[14] H.-H. Lin, T.-L. Liu, and J.-C. Chuang, “Learning a scene background model via classification,” IEEE Signal Process. Mag., vol. 57, no. 5, pp. 1641–1654, May 2009.

[15] L. Maddalena and A. Petrosino, “A self-organizing approach to background subtraction for visual surveillance applications,” IEEE Trans. Image Process., vol. 17, no. 7, pp. 1168–1177, Jul. 2008.

[16] V. Cevher, A. Sankaranarayanan, M. Duarte, D. Reddy, R. Baraniuk, and R. Chellappa, “Compressive sensing for background subtraction,” in Proc. Eur. Conf. Comput. Vis., Oct. 2008, pp. 155–168.

[17] M. Dikmen and T. Huang, “Robust estimation of foreground in surveillance videos by sparse error estimation,” in Proc. IEEE Int. Conf. Pattern Recognit., Tampa, FL, Dec. 2008, pp. 1–4.

[18] S. Cohen, “Background estimation as a labeling problem,” in Proc. Int. Conf. Comput. Vis., Beijing, China, Oct. 2005, vol. 2, pp. 1034–1041.

[19] V. Mahadevan and N. Vasconcelos, “Spatiotemporal saliency in dynamic scenes,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 32, no. 1, pp. 171–177, Jan. 2010.

[20] M. Sivabalakrishnan and D. Manjula, “An efficient foreground detection algorithm for visual surveillance system,” Int. J. Comput. Sci. Network Sec., vol. 9, pp. 221–227, May 2009.

[21] Cavallaro and T. Ebrahimi, “Video object extraction based on adaptive background and statistical change detection,” in Proc. Vis. Commun. Image Process., Jan. 2001, pp. 465–475.

[22] El Maadi and X. Maldague, “Outdoor infrared video surveillance: A novel dynamic technique for the subtraction of a changing background of IR images,” Infrared Phys. Technol., vol. 49, pp. 261–265, Jan. 2007.

[23] R. Abbott and L. Williams, “Multiple target tracking with lazy background subtraction and connected components analysis,” Mach. Vis. Appl., vol. 20, pp. 93–101, Feb. 2009.

[24] Shoushtarian and H. Bez, “A practical adaptive approach for dynamic background

subtraction using an invariant colour model and object tracking," Pattern Recognit. Lett., vol. 26, pp. 5–26, Jan. 2005.

[25] J. Cezar, C. Rosito, and S. Musse, "A background subtraction model adapted to illumination changes," in Proc. IEEE Int. Conf. Image Process., Oct. 2006, pp. 1817–1820.

[26] J. Davis and V. Sharma, "Robust background-subtraction for person detection in thermal imagery," in Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit., Washington, DC, Jun. 2004, vol. 8, p. 128.

[27] Wren, A. Azarbayejani, T. Darrell, and A. Pentland, "Pfinder: Real-time tracking of the human body," IEEE Trans. Pattern Anal. Mach. Intell., vol. 19, no. 7, pp. 780–785, Jul. 1997.

[28] K. Toyama, J. Krumm, B. Brumitt, and M. Meyers, "Wallflower: Principles and practice of background maintenance," in Proc. Int. Conf. Comput. Vis., Kerkyra, Greece, Sep. 1999, pp. 255–261.

[29] Koller, J. Weber, and J. Malik, "Robust multiple car tracking with occlusion reasoning," in Proc. Eur. Conf. Comput. Vis., Stockholm, Sweden, May 1994, pp. 189–196.

[30] J. Davis and V. Sharma, "Background-subtraction in thermal imagery using contour saliency," Int. J. Comput. Vis., vol. 71, pp. 161–181, Feb. 2007.

[31] C. Jung, "Efficient background subtraction and shadow removal for monochromatic video sequences," IEEE Trans. Multimedia, vol. 11, no. 3, pp. 571–577, Apr. 2009.

[32] Gutchess, M. Trajkovics, E. Cohen-Solal, D. Lyons, and A. Jain, "A background model initialization algorithm for video surveillance," in Proc. Int. Conf. Comput. Vis., Vancouver, BC, Jul. 2001, vol. 1, pp. 733–740.

[33] Haritaoglu, D. Harwood, and L. Davis, "$W^4$: Real-time surveillance of people and their activities," IEEE Trans. Pattern Anal. Mach. Intell., vol. 22, no. 8, pp. 809–830, Aug. 2000.

[34] Jacques, C. Jung, and S. Musse, "Background subtraction and shadow detection in grayscale video sequences," in Proc. Brazilian Symp. Comput. Graph. Image Process., Natal, Brazil, Oct. 2005, pp. 189–196.

[35] Manzanera and J. Richefeu, "A robust and computationally efficient motion detection algorithm based on sigma-delta background estimation," in Proc. Indian Conf. Comput. Vis., Graph. Image Process., Kolkata, India, Dec. 2004, pp. 46–51.

[36] Manzanera and J. Richefeu, “A new motion detection algorithmbased on $\sum - \Delta$ background estimation,” Pattern Recognit. Lett., vol. 28, pp. 320–328, Feb. 2007.

[37] Manzanera, “$\sum - \Delta$ background subtraction and the Zipf law,” in Proc. Progr. Pattern Recognit., Image Anal. Appl., Nov. 2007, pp. 42–51.

[38] Lacassagne, A. Manzanera, J. Denoulet, and A. Mérigot, “High performance motion detection: Some trends toward new embedded architectures for vision systems,” J. Real-Time Image Process., vol. 4, pp. 127–146, Jun. 2009.

[39] Lacassagne, A. Manzanera, and A. Dupret, “Motion detection: Fast and robust algorithms for embedded systems,” in Proc. IEEE Int. Conf. Image Process., Cairo, Egypt, Nov. 2009, pp. 3265–3268.

[40] Stauffer and E. Grimson, “Adaptive background mixture models for real-time tracking,” in Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit., Ft. Collins, CO, Jun. 1999, vol. 2, pp. 246–252.

[41] Stauffer and E. Grimson, “Learning patterns of activity using real-time tracking,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 22, no. 8, pp. 747–757, Aug. 2000.

[42] Lei and L. Xu, “Real-time outdoor video surveillance with robust foreground extraction and object tracking via multi-state transition management,” Pattern Recognit. Lett., vol. 27, pp. 1816–1825, Nov. 2006.

[43] Y. Wang, T. Tan, K. Loe, and J. Wu, “A probabilistic approach for foreground and shadow segmentation in monocular image sequences,” Pattern Recognit., vol. 38, pp. 1937–1946, Nov. 2005.

[44] Y. Wang, K. Loe, and J. Wu, “A dynamic conditional random field model for foreground and shadow segmentation,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 28, no. 2, pp. 279–289, Feb. 2006.

[45] Barnich, S. Jodogne, and M. Van Droogenbroeck, “Robust analysis of silhouettes by morphological size distributions,” in Advanced Concepts for Intelligent Vision Systems (ACIVS 2006), Vol. 4179 of Lecture Notes on Computer Science. New York: Springer-Verlag, Sep. 2006, pp. 734–745.

[46] J.-S. Hu and T.-M. Su, “Robust background subtraction with shadow and highlight removal

for indoor surveillance," EURASIP J. Appl. Signal Process., vol. 2007, pp. 108–108, Jan. 2007.

[47] P. KaewTraKulPong and R. Bowden, "An improved adaptive background mixture model for real-time tracking with shadow detection," in Proc. Eur. Workshop Adv. Video Based Surveillance Syst., London, U.K., Sep. 2001.

[48] Lee, "Effective Gaussian mixture learning for video background subtraction," IEEE Trans. Pattern Anal. Mach. Intell., vol. 27, no. 5, pp. 827–832, May 2005.

[49] Q. Zang and R. Klette, "Robust background subtraction and maintenance," in Proc. IEEE Int. Conf. Pattern Recognit., Washington, DC, Aug. 2004, vol. 2, pp. 90–93.

[50] Z. Zivkovic, "Improved adaptive gausian mixture model for background subtraction," in Proc. IEEE Int. Conf. Pattern Recognit., Cambridge, U.K., Aug. 2004, vol. 2, pp. 28–31.

[51] B. White and M. Shah, "Automatically tuning background subtraction parameters using particle swarm optimization," in Proc. IEEE Int. Conf. Multimedia Expo, Beijing, China, Jul. 2007, pp. 1826–1829.

[52] P. Varcheie, M. Sills-Lavoie, and G.-A. Bilodeau, "A multiscale region-based motion detection and background subtraction algorithm," Sensors, vol. 10, pp. 1041–1061, Jan. 2010.

[53] Elgammal, D. Harwood, and L. Davis, "Non-parametric model for background subtraction," in Proc. 6th Eur. Conf. Comput. Vis., London, U.K., Jun.–Jul. 2000, pp. 751–767.

[54] Srivastava, A. Lee, E. Simoncelli, and S.-C. Zhu, "On advances in statistical modeling of natural images," J. Math. Imag. Vis., vol. 18, pp. 17–33, Jan. 2003.

[55] Elgammal, R. Duraiswami, D. Harwood, and L. Davis, "Background and foreground modeling using nonparametric kernel density estimation for visual surveillance," Proc. IEEE, vol. 90, no. 7, pp. 1151–1163, Jul. 2002.

[56] Mittal and N. Paragios, "Motion-based background subtraction using adaptive kernel density estimation," in Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit., Los Alamitos, CA, Jun.–Jul. 2004, vol. 2, pp. 302–309.

[57] Y. Sheikh and M. Shah, "Bayesian modeling of dynamic scenes for object detection," IEEE Trans. Pattern Anal. Mach. Intell., vol. 27, no. 11, pp. 1778–1792, Nov. 2005.

[58] Z. Zivkovic and F. van der Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction," Pattern Recognit. Lett., vol. 27, pp. 773–780, May 2006.

[59] Tavakkoli, M. Nicolescu, G. Bebis, and M. Nicolescu, “Non-parametric statistical background modeling for efficient foreground region detection,” Mach. Vis. Appl., vol. 20, pp. 395–409, Oct. 2008.

[60] Monari and C. Pasqual, “Fusion of background estimation approaches for motion detection in non-static backgrounds,” in Proc. IEEE Int. Conf. Adv. Video Signal Based Surveillance, London, U.K., Sep. 2007, pp. 347–352.

[61] K. Kim, T. Chalidabhongse, D. Harwood, and L. Davis, “Background modeling and subtraction by codebook construction,” in Proc. IEEE Int. Conf. Image Process., Singapore, Oct. 2004, vol. 5, pp. 3061–3064.

[62] K. Kim, T. Chalidabhongse, D. Harwood, and L. Davis, “Real-time foreground-background segmentation using codebook model,” Real-Time Imag., vol. 11, Special Issue on Video Object Processing, pp. 172–185, Jun. 2005.

[63] Wu and X. Peng, “Spatio-temporal context for codebook-based dynamic background subtraction,” Int. J. Electron. Commun., vol. 64, no.8, pp. 739–747, 2010.

[64] H. Wang and D. Suter, “Background subtraction based on a robust consensus method,” in Proc. IEEE Int. Conf. Pattern Recognit., Washington, DC, Aug. 2006, pp. 223–226.

[65] H. Wang and D. Suter, “A consensus-based method for tracking: Modelling background scenario and foreground appearance,” Pattern Recognit., vol. 40, pp. 1091–1105, Mar. 2007.

[66] P.-M. Jodoin, M. Mignotte, and J. Konrad, “Statistical background subtraction using spatial cues,” IEEE Trans. Circuits Syst. Video Technol., vol. 17, no. 12, pp. 1758–1763, Dec. 2007.

[67] Criminisi, P. Pérez, and K. Toyama, “Region filling and object removal by exemplar-based image inpainting,” IEEE Trans. Image Process., vol. 13, no. 9, pp. 1200–1212, Sep. 2004.

[68] Papoulis, Probability, Random Variables, and Stochastic Processes. New York: McGraw-Hill, 1984.

[69] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, “Detecting moving objects, ghosts, and shadows in video streams,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 25, no. 10, pp. 1337–1342, Oct. 2003.

[70] L. Li, W. Huang, I. Gu, and Q. Tian, “Foreground object detection from videos containing complex background,” in Proc. ACM Int. Conf. Multimedia, Berkeley, CA, Nov. 2003, pp. 2–10.

[71] C.-C. Chiu, M.-Y. Ku, and L.-W. Liang, “A robust object segmentation system using a probability-based background extraction algorithm,” IEEE Trans. Circuits Syst. Video Technol., vol. 20, no. 4, pp. 518–528, Apr. 2010.

[72] Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in Proc. Int. Joint Conf. Artif. Intell., Vancouver, BC, Apr. 1981, pp. 674–679.

# ViBe: A Universal Background Subtraction Algorithm for Video Sequences

Olivier Barnich and Marc Van Droogenbroeck, *Member, IEEE*

***Abstract*—This paper presents a technique for motion detection that incorporates several innovative mechanisms. For example, our proposed technique stores, for each pixel, a set of values taken in the past at the same location or in the neighborhood. It then compares this set to the current pixel value in order to determine whether that pixel belongs to the background, and adapts the model by choosing randomly which values to substitute from the background model. This approach differs from those based upon the classical belief that the oldest values should be replaced first. Finally, when the pixel is found to be part of the background, its value is propagated into the background model of a neighboring pixel. We describe our method in full details (including pseudo-code and the parameter values used) and compare it to other background subtraction techniques. Efficiency figures show that our method outperforms recent and proven state-of-the-art methods in terms of both computation speed and detection rate. We also analyze the performance of a downscaled version of our algorithm to the absolute minimum of one comparison and one byte of memory per pixel. It appears that even such a simplified version of our algorithm performs better than mainstream techniques.**

***Index Terms*—Background subtraction, computer vision, image motion analysis, image segmentation, learning (artificial intelligence), pixel classification, real-time systems, surveillance, vision and scene understanding, video signal processing.**

## I. Introduction

THE number of cameras available worldwide has increased dramatically over the last decade. But this growth has resulted in a huge augmentation of data, meaning that the data are impossible either to store or to handle manually. In order to detect, segment, and track objects automatically in videos, several approaches are possible. Simple motion detection algorithms compare a static background frame with the current frame of a video scene, pixel by pixel. This is the basic principle of background subtraction, which can be formulated as a technique that builds a model of a background and compares this model with the current frame in order to detect zones where a significant difference occurs. The purpose of a background subtraction algorithm is, therefore, to distinguish moving objects (hereafter referred to as the *foreground*) from static, or slow moving, parts of the scene (called *background*). Note that when a static object starts moving, a background subtraction algorithm detects the object in motion as well as a hole left behind in the background (referred to as a *ghost*). Clearly a ghost is irrelevant for motion interpretation and has to be discarded. An alternative definition for the background is that it corresponds to a reference frame with values visible most of the time, that is with the highest appearance probability, but this kind of framework is not straightforward to use in practice.

While a static background model might be appropriate for analyzing short video sequences in a constrained indoor environment, the model is ineffective for most practical situations; a more sophisticated model is, therefore, required. Moreover, the detection of motion is often only a first step in the process of understanding the scene. For example, zones where motion is detected might be filtered and characterized for the detection of unattended bags, gait recognition, face detection, people counting, traffic surveillance, etc. The diversity of scene backgrounds and applications explains why countless papers discuss issues related to background subtraction.

In this paper, we present a universal method for background subtraction. This method has been briefly described in [1] and in a patent [2]. In Section II, we extensively review the literature of background subtraction algorithms. This review presents the major frameworks developed for background subtraction and highlights their respective advantages. We have implemented some of these algorithms in order to compare them with our method. Section III describes our technique and details our major innovations: the background model, the initialization process, and the update mechanism. Section IV discusses experimental results including comparisons with other state-of-the-art algorithms and computational performance. We also present a simplified version of our algorithm which requires only one comparison and one byte of memory per pixel; this is the absolute minimum in terms of comparisons and memory for any background subtraction technique. We show that, even in its simplified form, our algorithm performs better than more sophisticated techniques. Section V concludes the paper.

## II. Review of Background Subtraction Algorithms

The problem tackled by background subtraction techniques involves the comparison of an observed image with an estimated image that does not contain any object of interest; this is referred to as the background model (or background image) [3]. This comparison process, called *foreground detection*, divides the observed image into two complementary sets of pixels that cover the entire image: 1) the foreground that contains the objects of interest, and 2) the background, its complementary set.

Manuscript received June 01, 2010; revised August 20, 2010, November 22, 2010; accepted December 01, 2010. Date of publication December 23, 2010; date of current version May 18, 2011. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Mark (Hong-Yuan) Liao.

O. Barnich is with EVS Broadcast Equipment, Liege Science Park, B-4102 Seraing, Belgium (e-mail: olivier.barnich@gmail.com).

M. Van Droogenbroeck is with the Faculty of Applied Sciences, University of Liège, Liège, B-4000 Belgium (e-mail: m.vandroogenbroeck@ulg.ac.be).

Digital Object Identifier 10.1109/TIP.2010.2101613

As stated in [4], it is difficult to specify a gold-standard definition of what a background subtraction technique should detect as a foreground region, as the definition of foreground objects relates to the application level.

Many background subtraction techniques have been proposed with as many models and segmentation strategies, and several surveys are devoted to this topic (see for example [3]–[9]). Some algorithms focus on specific requirements that an ideal background subtraction technique could or should fulfill. According to [7], a background subtraction technique must adapt to gradual or fast illumination changes (changing time of day, clouds, etc), motion changes (camera oscillations), high frequency background objects (e.g., tree leave or branches), and changes in the background geometry (e.g., parked cars). Some applications require background subtraction algorithms to be embedded in the camera, so that the computational load becomes the major concern. For the surveillance of outdoor scenes, robustness against noise and adaptivity to illumination changes are also essential.

Most techniques described in the literature operate on each pixel independently. These techniques relegate entirely to post-processing algorithms the task of adding some form of spatial consistency to their results. Since perturbations often affect individual pixels, this results in local misclassifications. By contrast, the method described by Seiki *et al.* in [10] is based upon the assumption that neighboring blocks of background pixels should follow similar variations over time. While this assumption holds most of the time, especially for pixels belonging to the same background object, it becomes problematic for neighboring pixels located at the border of multiple background objects. Despite this inconvenience, pixels are aggregated into blocks and each $N \times N$ block is processed as an $N^2$-component vector. A few samples are then collected over time and used to train a principal component analysis (PCA) model for each block. A block of a new video frame is classified as background if its observed image pattern is close to its reconstructions using PCA projection coefficients of 8-neighboring blocks. Such a technique is also described in [11], but it lacks an update mechanism to adapt the block models over time. In [12], the authors focus on the PCA reconstruction error. While the PCA model is also trained with time samples, the resulting model accounts for the whole image. Individual pixels are classified as background or foreground using simple image difference thresholding between the current image and the back-projection in the image space of its PCA coefficients. As for other PCA-based methods, the initialization process and the update mechanism are not described.

A similar approach, the independent component analysis (ICA) of serialized images from a training sequence, is described in [13] in the training of an ICA model. The resulting demixing vector is then computed and compared to that of a new image in order to separate the foreground from a reference background image. The method is said to be highly robust to indoor illumination changes.

A two-level mechanism based upon a classifier is introduced in [14]. A classifier first determines whether an image block belongs to the background. Appropriate blockwise updates of the background image are then carried out in the second stage, depending upon the results of the classification. Classification algorithms are also the basis of other algorithms, as in the one provided in [15], where the background model learns its motion patterns by self organization through artificial neural networks.

Algorithms based upon the framework of compressive sensing perform background subtraction by learning and adapting a low-dimensional compressed representation of the background [16]. The major advantage of this approach lies in the fact that compressive sensing estimates object silhouettes without any auxiliary image reconstruction. On the other hand, objects in the foreground need to occupy only a small portion of the camera view in order to be detected correctly.

Background subtraction is considered to be a sparse error recovery problem in [17]. These authors assumed that each color channel in the video can be independently modeled as the linear combination of the same color channel from other video frames. Consequently, the method they proposed is able to accurately compensate for global changes in the illumination sources without altering the general structure of the frame composition by finding appropriate scalings for each color channel separately.

Background estimation is formulated in [18] as an optimal labeling problem in which each pixel of the background image is labeled with a frame number, indicating which color from the past must be copied. The author's proposed algorithm produces a background image, which is constructed by copying areas from the input frames. Impressive results are shown for static backgrounds but the method is not designed to cope with objects moving slowly in the background, as its outcome is a single static background frame.

The authors of [19] were inspired by the biological mechanism of motion-based perceptual grouping. They propose a spatio-temporal saliency algorithm applicable to scenes with highly dynamic backgrounds, which can be used to perform background subtraction. Comparisons of their algorithm with other state-of-the-art techniques show that their algorithm reduces the average error rate, but at a cost of a prohibitive processing time (several seconds per frame), which makes it unsuitable for real-time applications.

Pixel-based background subtraction techniques compensate for the lack of spatial consistency by a constant updating of their model parameters. The simplest techniques in this category are the use of a static background frame (which has recently been used in [20]), the (weighted) running average [21], first-order low-pass filtering [22], temporal median filtering [23], [24], and the modeling of each pixel with a Gaussian [25]–[27].

Probabilistic methods predict the short-term evolution of a background frame with a Wiener [28] or a Kalman [29] filter. In [28], a frame-level component is added to the pixel-level operations. Its purpose is to detect sudden and global changes in the image and to adapt the background frame accordingly. Median and Gaussian models can be combined to allow inliers (with respect to the median) to have more weight than outliers during the Gaussian modeling, as in [30] or [31]. A method for properly initializing a Gaussian background model from a video sequence in which moving objects are present is proposed in [32].

The $W^4$ model presented in [33] is a rather simple but nevertheless effective method. It uses three values to represent each

pixel in the background image: the minimum and maximum intensity values, and the maximum intensity difference between consecutive images of the training sequence. The authors of [34] bring a small improvement to the $W^4$ model together with the incorporation of a technique for shadow detection and removal.

Methods based upon $\Sigma$-$\Delta$ (sigma-delta) motion detection filters [35]–[37] are popular for embedded processing [38], [39]. As in the case of analog-to-digital converters, a $\Sigma$-$\Delta$ motion detection filter consists of a simple nonlinear recursive approximation of the background image, which is based upon comparison and on an elementary increment/decrement (usually $-1$, 0, and 1 are the only possible updating values). The $\Sigma$-$\Delta$ motion detection filter is, therefore, well suited to many embedded systems that lack a floating point unit.

All these unimodal techniques can lead to satisfactory results in controlled environments while remaining fast, easy to implement, and simple. However, more sophisticated methods are necessary when dealing with videos captured in complex environments where moving background, camera egomotion, and high sensor noise are encountered [5].

Over the years, increasingly complex pixel-level algorithms have been proposed. Among these, by far the most popular is the Gaussian Mixture Model (GMM) [40], [41]. First presented in [40], this model consists of modeling the distribution of the values observed over time at each pixel by a weighted mixture of Gaussians. This background pixel model is able to cope with the multimodal nature of many practical situations and leads to good results when repetitive background motions, such as tree leaves or branches, are encountered. Since its introduction, the model has gained vastly in popularity among the computer vision community [4], [7], [11], [42]–[44], and it is still raising a lot of interest as authors continue to revisit the method and propose enhanced algorithms [45]–[50]. In [51], a particle swarm optimization method is proposed to automatically determine the parameters of the GMM algorithm. The authors of [52] combine a GMM model with a region-based algorithm based upon color histograms and texture information. In their experiments, the authors' method outperform the original GMM algorithm. However, the authors' technique has a considerable computational cost as they only manage to process seven frames of $640 \times 480$ pixels per second with an Intel Xeon 5150 processor.

The downside of the GMM algorithm resides in its strong assumptions that the background is more frequently visible than the foreground and that its variance is significantly lower. None of this is valid for every time window. Furthermore, if high- and low-frequency changes are present in the background, its sensitivity cannot be accurately tuned and the model may adapt to the targets themselves or miss the detection of some high speed targets, as detailed in [53]. Also, the estimation of the parameters of the model (especially the variance) can become problematic in real-world noisy environments. This often leaves one with no other choice than to use a fixed variance in a hardware implementation. Finally, it should be noted that the statistical relevance of a Gaussian model is debatable as some authors claim that natural images exhibit non-Gaussian statistics [54].

To avoid the difficult question of finding an appropriate shape for the probability density function, some authors have turned their attention to nonparametric methods to model background distributions. One of the strengths of nonparametric kernel density estimation methods [53], [55]–[59] is their ability to circumvent a part of the delicate parameter estimation step due to the fact that they rely on pixel values observed in the past. For each pixel, these methods build a histogram of background values by accumulating a set of real values sampled from the pixel's recent history. These methods then estimate the probability density function with this histogram to determine whether or not a pixel value of the current frame belongs to the background. Nonparametric kernel density estimation methods can provide fast responses to high-frequency events in the background by directly including newly observed values in the pixel model. However, the ability of these methods to successfully handle concomitant events evolving at various speeds is questionable since they update their pixel models in a first-in first-out manner. This has led some authors to represent background values with two series of values or models: a short term model and a long term model [53], [60]. While this can be a convenient solution for some situations, it leaves open the question of how to determine the proper time interval. In practical terms, handling two models increases the difficulty of fine-tuning the values of the underlying parameters. Our method incorporates a smoother lifespan policy for the sampled values, and as explained in Section IV, it improves the overall detection quality significantly.

In the codebook algorithm [61], [62], each pixel is represented by a codebook, which is a compressed form of background model for a long image sequence. Each codebook is composed of codewords comprising colors transformed by an innovative color distortion metric. An improved codebook incorporating the spatial and temporal context of each pixel has been proposed in [63]. Codebooks are believed to be able to capture background motion over a long period of time with a limited amount of memory. Therefore, codebooks are learned from a typically long training sequence and a codebook update mechanism is described in [62] allowing the algorithm to evolve with the lighting conditions once the training phase is over. However, one should note that the proposed codebook update mechanism does not allow the creation of new codewords, and this can be problematic if permanent structural changes occur in the background (for example, in the case of newly freed parking spots in urban outdoor scenes).

Instead of choosing a particular form of background density model, the authors of [64], [65] use the notion of "consensus." They keep a cache of a given number of last observed background values for each pixel and classify a new value as background if it matches most of the values stored in the pixel's model. One might expect that such an approach would avoid the issues related to deviations from an arbitrarily assumed density model, but since values of pixel models are replaced according to a first-in first-out update policy, they are also prone to the problems discussed previously, for example, the problem of slow and fast motions in the background, unless a large number of pixel samples are stored. The authors state that a cache of 20 samples is the minimum required for the method to be useful, but they also noticed no significant further improvement for caches with more than 60 samples. Consequently, the training period for their algorithm must comprise at least 20 frames. Finally, to cope with lighting changes and objects appearing or

fading in the background, two additional mechanisms (one at the pixel level, a second at the blob level) are added to the consensus algorithm to handle entire objects.

The method proposed in this paper operates differently in handling new or fading objects in the background, without the need to take account of them explicitly. In addition to being faster, our method exhibits an interesting asymmetry in that a ghost (a region of the background discovered once a static object starts moving) is added to the background model more quickly than an object that stops moving. Another major contribution of this paper resides in the proposed update policy. The underlying idea is to gather samples from the past and to update the sample values by ignoring when they were added to the models. This policy ensures a smooth exponential decaying lifespan for the sample values of the pixel models and allows our technique to deal with concomitant events evolving at various speeds with a unique model of a reasonable size for each pixel.

## III. Description of a Universal Background Subtraction Technique: ViBe

Background subtraction techniques have to deal with at least three considerations in order to be successful in real applications: 1) what is the model and how does it behave? 2) how is the model initialized? and 3) how is the model updated over time? Answers to these questions are given in the three subsections of this section. Most papers describe the intrinsic model and the updating mechanism. Only a minority of papers discuss initialization, which is critical when a fast response is expected, as in the case inside a digital camera. In addition, there is often a lack of coherence between the model and the update mechanism. For example, some techniques compare the current value of a pixel $p$ to that of a model $b$ with a given tolerance $T$. They consider that there is a good match if the absolute difference between $p$ and $b$ is lower than $T$. To be adaptive over time, $T$ is adjusted with respect to the statistical variance of $p$. But the statistical variance is estimated by a temporal average. Therefore, the adjustment speed is dependent upon the acquisition framerate and on the number of background pixels. This is inappropriate in some cases, as in the case of remote IP cameras whose framerate is determined by the available bandwidth.

We detail in the following a background subtraction technique, called visual background extractor (ViBe). For convenience, we present a complete version of our algorithm in a C-like code in Appendix A.

### A. Pixel Model and Classification Process

To some extent, there is no way around the determination, for a given color space, of a probability density function (pdf) for every background pixel or at least the determination of statistical parameters, such as the mean or the variance. Note that with a Gaussian model, there is no distinction to be made as the knowledge of the mean and variance is sufficient to determine the pdf. While the classical approaches to background subtraction and most mainstream techniques rely on pdfs or statistical parameters, the question of their statistical significance is rarely discussed, if not simply ignored. In fact, there is no imperative to compute the pdf as long as the goal of reaching a relevant background segmentation is achieved. An alternative is to consider that one should enhance statistical significance over time, and one way to proceed is to build a model with real observed pixel values. The underlying assumption is that this makes more sense from a stochastic point of view, as already observed values should have a higher probability of being observed again than would values not yet encountered.

Like the authors of [65], we do not opt for a particular form for the pdf, as deviations from the assumed pdf model are ubiquitous. Furthermore, the evaluation of the pdf is a global process and the shape of a pdf is sensitive to outliers. In addition, the estimation of the pdf raises the nonobvious question regarding the number of samples to be considered; the problem of selecting a representative number of samples is intrinsic to all the estimation processes.

If we see the problem of background subtraction as a classification problem, we want to classify a new pixel value with respect to its immediate neighborhood in the chosen color space, so as to avoid the effect of any outliers. This motivates us to model each background pixel with a set of samples instead of with an explicit pixel model. Consequently no estimation of the pdf of the background pixel is performed, and so the current value of the pixel is compared to its closest samples within the collection of samples. This is an important difference in comparison with existing algorithms, in particular with those of consensus-based techniques. A new value is compared to background samples and should be close to some of the sample values instead of the majority of all values. The underlying idea is that it is more reliable to estimate the statistical distribution of a background pixel with a small number of close values than with a large number of samples. This is somewhat similar to ignoring the extremities of the pdf, or to considering only the central part of the underlying pdf by thresholding it. On the other hand, if one trusts the values of the model, it is crucial to select background pixel samples carefully. The classification of pixels in the background, therefore, needs to be conservative, in the sense that only background pixels should populate the background models.

Formally, let us denote by $v(x)$ the value in a given Euclidean color space taken by the pixel located at $x$ in the image, and by $v_i$ a background sample value with an index $i$. Each background pixel $x$ is modeled by a collection of $N$ background sample values

$$\mathcal{M}(x) = \{v_1, v_2, \ldots, v_N\} \tag{1}$$

taken in previous frames. For now, we ignore the notion of time; this is discussed later.

To classify a pixel value $v(x)$ according to its corresponding model $\mathcal{M}(x)$, we compare it to the *closest* values within the set of samples by defining a sphere $S_R(v(x))$ of radius $R$ centered on $v(x)$. The pixel value $v(x)$ is then classified as background if the cardinality, denoted $\sharp$, of the set intersection of this sphere and the collection of model samples $\mathcal{M}(x)$ is larger than or equal to a given threshold $\sharp_{\min}$. More formally, we compare $\sharp_{\min}$ to

$$\sharp\{S_R(v(x)) \cap \{v_1, v_2, \ldots, v_N\}\}. \tag{2}$$

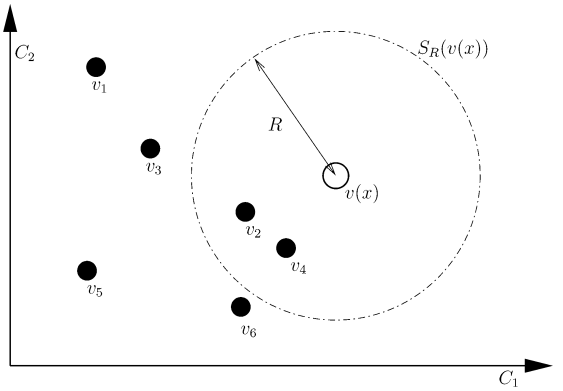


Fig. 1. Comparison of a pixel value with a set of samples in a 2-D Euclidean color space $(C_1, C_2)$. To classify $v(x)$, we count the number of samples of $\mathcal{M}(x)$ intersecting the sphere of radius $R$ centered on $v(x)$.

According to (2), the classification of a pixel value $v(x)$ involves the computation of $N$ distances between $v(x)$ and model samples, and of $N$ comparisons with a thresholded Euclidean distance $R$. This process is illustrated in Fig. 1. Note that, as we are only interested in finding a few matches, the segmentation process of a pixel can be stopped once $\sharp_{\min}$ matches have been found.

As can easily be seen, the accuracy of our model is determined by two parameters only: the radius $R$ of the sphere and the minimal cardinality $\sharp_{\min}$. Experiments have shown that a unique radius R of 20 (for monochromatic images) and a cardinality of 2 are appropriate (see Section IV-A for a thorough discussion on parameter values). There is no need to adapt these parameters during the background subtraction nor do we need to change them for different pixel locations within the image. Note that since the number of samples $N$ and $\sharp_{\min}$ are chosen to be fixed and since they impact on the same decision, the sensitivity of the model can be adjusted using the following ratio:

$$\frac{\sharp_{\min}}{N} \tag{3}$$

but in all our comparative tests we kept these values unchanged.

So far we have detailed the nature of the model. In the coming sections, we explain how to initialize the model from a single frame and how to update it over time.

### B. Background Model Initialization From a Single Frame

Many popular techniques described in the literature, such as [53], [62], and [65], need a sequence of several dozens of frames to initialize their models. Such an approach makes sense from a statistical point of view as it seems imperative to gather a significant amount of data in order to estimate the temporal distribution of the background pixels. But one may wish to segment the foreground of a sequence that is even shorter than the typical initialization sequence required by some background subtraction algorithms. Furthermore, many applications require the ability to provide an *uninterrupted* foreground detection, even in the presence of sudden light changes, which cannot be handled properly by the regular update mechanism of the algorithm. A possible solution to both these issues is to provide a specific model update process that tunes the pixel models to new lighting conditions. But the use of such a dedicated update process is at best delicate, since a sudden illumination may completely alter the chromatic properties of the background.

A more convenient solution is to provide a technique that will initialize the background model from a single frame. Given such a technique, the response to sudden illumination changes is straightforward: the existing background model is discarded and a new model is initialized instantaneously. Furthermore, being able to provide a reliable foreground segmentation as early on as the second frame of a sequence has obvious benefits for short sequences in video-surveillance or for devices that embed a motion detection algorithm.

Since there is no temporal information in a single frame, we use the same assumption as the authors of [66], which is that neighboring pixels share a similar temporal distribution. This justifies the fact that we populate the pixel models with values found in the spatial neighborhood of each pixel. More precisely, we fill them with values randomly taken in their neighborhood in the first frame. The size of the neighborhood needs to be chosen so that it is large enough to comprise a sufficient number of different samples, while keeping in mind that the statistical correlation between values at different locations decreases as the size of the neighborhood increases. From our experiments, selecting samples randomly in the 8-connected neighborhood of each pixel has proved to be satisfactory for images of $640 \times 480$ pixels.

Formally, we assume that $t = 0$ indexes the first frame and that $N_G(x)$ is a spatial neighborhood of a pixel location $x$, therefore

$$\mathcal{M}^0(x) = \{v^0(y \,|\, y \in N_G(x))\} \tag{4}$$

where locations $y$ are chosen randomly according to a uniform law. Note that it is possible for a given $v^0(y)$ to be selected several times (for example if the size of the neighborhood is smaller than the cardinality of $\mathcal{M}^0(x)$) or to not be selected at all. However, this is not an issue if one acknowledges that values in the neighborhood are excellent sample candidates.

This strategy has proved to be successful. The only drawback is that the presence of a moving object in the first frame will introduce an artifact commonly called a *ghost*. According to [24], a ghost is "a set of connected points, detected as in motion but not corresponding to any real moving object." In this particular case, the ghost is caused by the unfortunate initialization of pixel models with samples coming from the moving object. In subsequent frames, the object moves and uncovers the real background, which will be learned progressively through the regular model update process, making the ghost fade over time. Fortunately, as shown in Section IV-C, our update process ensures both a fast model recovery in the presence of a ghost and a slow incorporation of real moving objects into the background model.

### C. Updating the Background Model Over Time

In this section, we describe how to continuously update the background model with each new frame. This a crucial step if we want to achieve accurate results over time: the update process must be able to adapt to lighting changes and to handle new objects that appear in a scene.

*1) General Discussions on an Update Mechanism:* The classification step of our algorithm compares the current pixel value $v^t(x)$ directly to the samples contained in the background model of the previous frame, $\mathcal{M}^{t-1}(x)$ at time $t-1$. Consequently, the question regarding *which* samples have to be memorized by the model and for *how long* is essential. One can see the model as a background memory or background history, as it is often referred to in the literature. The classical approach to the updating of the background history is to discard and replace old values after a number of frames or after a given period of time (typically about a few seconds); the oldest values are substituted by the new ones. Despite the rationale behind it, this substitution principle is not so obvious, as there is no reason to remove a valid value if it corresponds to a background value.

The question of including or not foreground pixel values in the model is one that is always raised for a background subtraction method based upon samples; otherwise the model will not adapt to changing conditions. It boils down to a choice between a conservative and a blind update scheme. Note that kernel-based pdf estimation techniques have a softer approach to updating. They are able to smooth the appearance of a new value by giving it a weight prior to inclusion.

*A conservative update* policy never includes a sample belonging to a foreground region in the background model. In practice, a pixel sample can be included in the background model only if it has been classified as a background sample. Such a policy seems, at first sight, to be the obvious choice. It actually guarantees a sharp detection of the moving objects, given that they do not share similar colors with the background. Unfortunately, it also leads to deadlock situations and everlasting ghosts: a background sample incorrectly classified as foreground prevents its background pixel model from being updated. This can keep indefinitely the background pixel model from being updated and could cause a permanent misclassification. Unfortunately, many practical scenarios lead to such situations. For example, the location freed by a previously parked car cannot be included in the background model with a purely conservative update scheme, unless a dedicated update mechanism handles such situations.

*Blind update* is not sensitive to deadlocks: samples are added to the background model whether they have been classified as background or not. The principal drawback of this method is a poor detection of slow moving targets, which are progressively included in the background model. A possible solution consists of using pixel models of a large size, which cover long time windows. But this comes at the price of both an increased memory usage and a higher computational cost. Furthermore, with a first-in first-out model update policy such as those employed in [53] or [65], 300 samples cover a time window of only 10 s [at 30 frames per second (FPS)]. A pixel covered by a slowly moving object for more than 10 s would still be included in the background model.

Strictly speaking, temporal information is not available when the background is masked. But background subtraction is a spatio-temporal process. In order to improve the technique, we could assume, as we proposed in Section III-B, that neighboring pixels are expected to have a similar temporal distribution. According to this hypothesis, the best strategy is, therefore, to adopt a conservative update scheme and to exploit spatial information in order to inject information regarding the background evolution into the background pixel models masked locally by the foreground. This process is common in inpainting, where objects are removed and values are taken in the neighborhood to fill holes [67]. In Section III-C.4, we provide a simple but effective method for exploiting spatial information, which enables us to counter most of the drawbacks of a purely conservative update scheme.

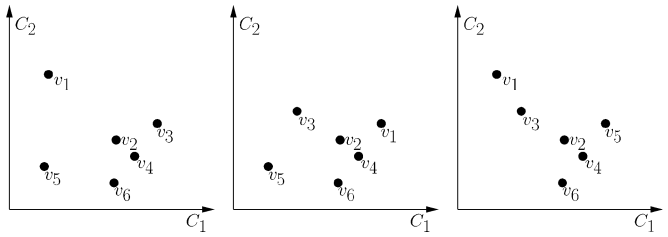


Fig. 2. Three of the six possible outcomes of the updating of a pixel model of size $N = 6$. We assume that values occupy the same color space as in Fig. 1 and that we have decided to update the model. This figure shows three possible models after the update. The decision process for selecting one particular model is random (with equal probabilities).

Our update method incorporates three important components: 1) a memoryless update policy, which ensures a smooth decaying lifespan for the samples stored in the background pixel models, 2) a random time subsampling to extend the time windows covered by the background pixel models, and 3) a mechanism that propagates background pixel samples spatially to ensure spatial consistency and to allow the adaptation of the background pixel models that are masked by the foreground. These components are described, together with our reasons for using them, in the following three subsections.

*2) A Memoryless Update Policy:* Many sample-based methods use first-in first-out policies to update their models. In order to deal properly with wide ranges of events in the scene background, Wang *et al.* [65] propose the inclusion of large numbers of samples in pixel models. But as stated earlier, this may still not be sufficient for high framerates. Other authors [53], [58] incorporate two temporal submodels to handle both fast and slow modifications. This approach proved to be effective. However, it increases the parametrization problem, in that it makes necessary to determine a greater number of parameter values in order to achieve a practical implementation.

From a theoretical point of view, we believe that it is more appropriate to ensure a monotonic decay of the probability of a sample value to remain inside the set of samples. A pixel model should contain samples from the recent past of the pixel but older samples should not necessarily be discarded.

We propose a method that offers an exponential monotonic decay for the remaining lifespan of the samples. The method improves the time relevance of the estimation by allowing a few old samples to remain in the pixel model. Remember that this approach is combined with a conservative update policy, so that foreground values should never be included in the models.

The technique, illustrated in Fig. 2, is simple but effective: instead of systematically removing the oldest sample from the pixel model, we choose the sample to be discarded randomly according to a uniform probability density function.

The new value then replaces the selected sample. This random strategy contradicts the idea that older values should be replaced first, which is not true for a conservative update policy. A conservative update policy is also necessary for the stability of the process. Indeed, the random update policy produces a nondeterministic background subtraction algorithm (to our knowledge, this is the first background subtraction algorithm to have that property). Only a conservative update policy ensures that the models do not diverge over time. Despite this, there may be slight differences, imperceptible in our experience, between the results of the same sequence processed by our background subtraction algorithm at different times.

Mathematically, the probability of a sample present in the model at time $t$ being preserved after the update of the pixel model is given by $(N-1)/(N)$. Assuming time continuity and the absence of memory in the selection procedure, we can derive a similar probability, denoted $P(t,\, t+dt)$ hereafter, for any further time $t+dt$. This probability is equal to

$$P(t,\, t+dt) = \left(\frac{N-1}{N}\right)^{(t+dt)-t} \tag{5}$$

which can be rewritten as

$$P(t,\, t+dt) = e^{-\ln(\frac{N}{N-1})dt}. \tag{6}$$

This expression shows that the expected remaining lifespan of any sample value of the model decays exponentially. It appears that the probability of a sample being preserved for the interval $(t,\, t+dt)$, assuming that it was included in the model prior to time $t$, is independent of $t$. In other words, the past has no effect on the future. This property, called the *memoryless* property, is known to be applicable to an exponential density (see [68]). This is a remarkable and, to our knowledge, unique property in the field of background subtraction. It completely frees us to define a time period for keeping a sample in the history of a pixel and, to some extent, allows the update mechanism to adapt to an arbitrary framerate.

*3) Time Subsampling:* We have shown how the use of a random replacement policy allow our pixel model to cover a large (theoretically infinite) time window with a limited number of samples. In order to further extend the size of the time window covered by a pixel model of a fixed size, we resort to random time subsampling. The idea is that in many practical situations, it is not necessary to update each background pixel model for each new frame. By making the background update less frequent, we artificially extend the expected lifespan of the background samples. But in the presence of periodic or pseudo-periodic background motions, the use of fixed subsampling intervals might prevent the background model from properly adapting to these motions. This motivates us to use a *random* subsampling policy. In practice, when a pixel value has been classified as belonging to the background, a random process determines whether this value is used to update the corresponding pixel model.

In all our tests, we adopted a time subsampling factor, denoted $\phi$, of 16: a background pixel value has one chance in 16 of being selected to update its pixel model. But one may wish to tune this parameter to adjust the length of the time window covered by the pixel model.

*4) Spatial Consistency Through Background Samples Propagation:* Since we use a conservative update scheme, we have to provide a way of updating the background pixel models that are hidden by the foreground. A popular way of doing this is to use what the authors of the $W^4$ algorithm [33] call a "*detection support map*" which counts the number of consecutive times that a pixel has been classified as foreground. If this number reaches a given threshold for a particular pixel location, the current pixel value at that location is inserted into the background model. A variant consists of including, in the background, groups of connected foreground pixels that have been found static for a long time, as in [69]. Some authors, like those of the $W^4$ algorithm and those of the SACON model [64], [65], use a combination of a pixel-level and an object-level background update.

The strength of using a conservative update comes from the fact that pixel values classified as foreground are never included in any background pixel model. While convenient, the support map related methods only delay the inclusion of foreground pixels. Furthermore, since these methods rely on a binary decision, it takes time to recover from a improper inclusion of a genuine foreground object in the background model. A progressive inclusion of foreground samples in the background pixel models is more appropriate.

As stated earlier, we have a different approach. We consider that neighboring background pixels share a similar temporal distribution and that a new background sample of a pixel should also update the models of neighboring pixels. According to this policy, background models hidden by the foreground will be updated with background samples *from neighboring pixel locations* from time to time. This allows a spatial diffusion of information regarding the background evolution that relies on samples classified *exclusively* as background. Our background model is, thus, able to adapt to a changing illumination and to structural evolutions (added or removed background objects) while relying on a *strict* conservative update scheme.

More precisely, let us consider the 4- or 8-connected spatial neighborhood of a pixel $x$, that is $N_G(x)$, and assume that it has been decided to update the set of samples $\mathcal{M}(x)$ by inserting $v(x)$. We then also use this value $v(x)$ to update the set of samples $\mathcal{M}(y \in N_G(x))$ from one of the pixels in the neighborhood, chosen at random according to a uniform law.

Since pixel models contain many samples, irrelevant information that could accidentally be inserted into the neighborhood model does not affect the accuracy of the detection. Furthermore, the erroneous diffusion of irrelevant information is blocked by the need to match an observed value before it can propagate further. This natural limitation inhibits the diffusion of error.

Note that neither the selection policy nor the spatial propagation method is deterministic. As stated earlier, if the algorithm is run over the same video sequence again, the results will always differ slightly (see Fig. 2). Although unusual, the strategy of allowing a random process to determine which samples are to be discarded proves to be very powerful. This is different from known strategies that introduce a fading factor or that use a long term and a short term history of values.

This concludes the description of our algorithm. ViBe makes no assumption regarding the video stream framerate or color space, nor regarding the scene content, the background itself, or its variability over time. Therefore, we refer to it as a universal method.

## IV. Experimental Results

In this section, we determine optimal values for the parameters of ViBe, and compare its results with those of seven other algorithms: two simple methods and five state-of-the art techniques. We also describe some advantageous and intrinsic properties of ViBe, and finally, we illustrate the suitability of ViBe for embedded systems.

For the sake of comparison, we have produced manually ground-truth segmentation maps for subsets of frames taken from two test sequences. The first sequence (called "house") was captured outdoor on a windy day. The second sequence ("pets") was extracted from the PETS2001 public data-set (data-set 3, camera 2, testing). Both sequences are challenging as they feature background motion, moving trees and bushes, and changing illumination conditions. The "pets" sequence is first used below to determine objective values for some of the parameters of ViBe. We then compare ViBe with seven existing algorithms on both sequences.

Many metrics can be used to assess the output of a background subtraction algorithm given a series of ground-truth segmentation maps. These metrics usually involve the following quantities: the number of true positives (TP), which counts the number of correctly detected foreground pixels; the number of false positives (FP), which counts the number of background pixels incorrectly classified as foreground; the number of true negatives (TN), which counts the number of correctly classified background pixels; and the number of false negatives (FN), which accounts for the number of foreground pixels incorrectly classified as background.

The difficulty of assessing background subtraction algorithms originates from the lack of a standardized evaluation framework; some frameworks have been proposed by various authors but mainly with the aim of pointing out the advantages of their own method. According to [6], the metric most widely used in computer vision to assess the performance of a binary classifier is the percentage of correct classification (PCC), which combines all four values

$$\mathrm{PCC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}. \tag{7}$$

This metric was adopted for our comparative tests. Note that the PCC percentage needs to be as high as possible, in order to minimize errors.

### A. Determination of Our Own Parameters

From previous discussions, it appears that ViBe has the following parameters:

- the radius $R$ of the sphere used to compare a new pixel value to pixel samples [see (2)];
- the time subsampling factor $\phi$;
- the number $N$ of samples stored in each pixel model;
- and the number $\sharp_{\min}$ of close pixel samples needed to classify a new pixel value as background [see (2)].

In our experience, the use of a radius $R = 20$ and a time subsampling factor $\phi = 16$ leads to excellent results in every situation. Note that the use of $R = 20$ is an educated choice, which corresponds to a perceptible difference in color.

To determine an optimal value for $\sharp_{\min}$, we compute the evolution of the PCC of ViBe on the "pets" sequence for $\sharp_{\min}$ ranging from 1 to 20. The other parameters were fixed to $N = 20$, $R = 20$, and $\phi = 16$. Fig. 3 shows that the best PCCs are obtained for $\sharp_{\min} = 2$ and $\sharp_{\min} = 3$.

Since a rise in $\sharp_{\min}$ is likely to increase the computational cost of ViBe, we set the optimal value of $\sharp_{\min}$ to $\sharp_{\min} = 2$. Note that in our experience, the use of $\sharp_{\min} = 1$ can lead to excellent results in scenes with a stable background.

Once the value of 2 has been selected for $\sharp_{\min}$, we study the influence of the parameter $N$ on the performance of ViBe. Fig. 4 shows percentages obtained on the "pets" sequence for $N$ ranging from 2 to 50 ($R$ and $\phi$ were set to 20 and 16). We observe that higher values of $N$ provide a better performance. However, they tend to saturate for values higher than 20. Since as for $\sharp_{\min}$, large $N$ values induce a greater computational cost, we select $N$ at the beginning of the plateau, that is $N = 20$.

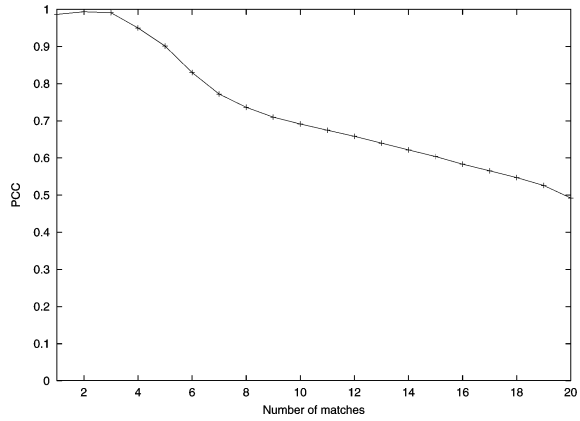


Fig. 3. PCCs for $\sharp_{\min}$ ranging from 1 to 20. The other parameters of ViBe were set to $N = 20$, $R = 20$, and $\phi = 16$.

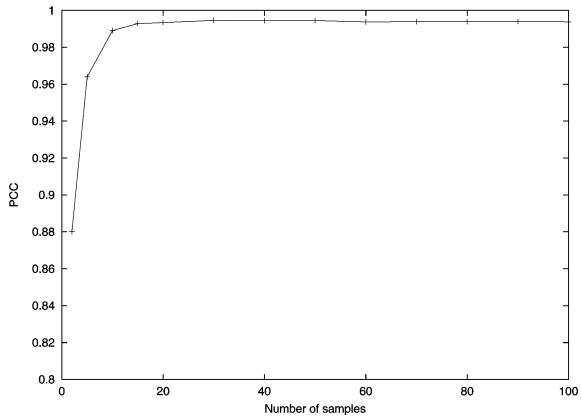


Fig. 4. PCCs given the number of samples collected in a background model.

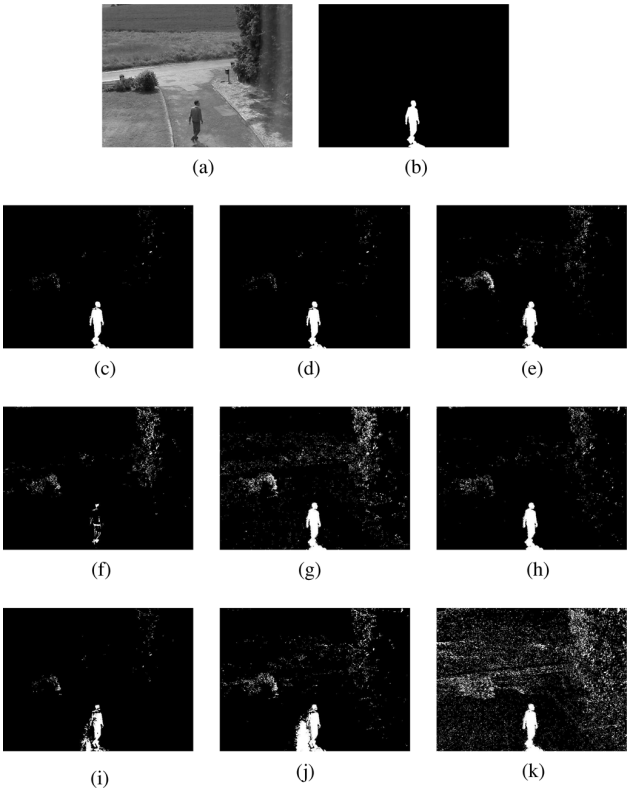


Fig. 5. Comparative background/foreground segmentation maps of nine background subtraction techniques for one frame taken from the "house" sequence. The segmentation maps of ViBe are the closest to the ground-truth reference. (a) Input image. (b) Ground-truth. (c) ViBe (RGB). (d) ViBe (gray). (e) Bayesian histogram. (f) Codebook. (g) EGMM [Zivkovic]. (h) GMM [Li *et al.*]. (i) Gaussian model. (j) First-order filter. (k) Sigma-Delta Z.

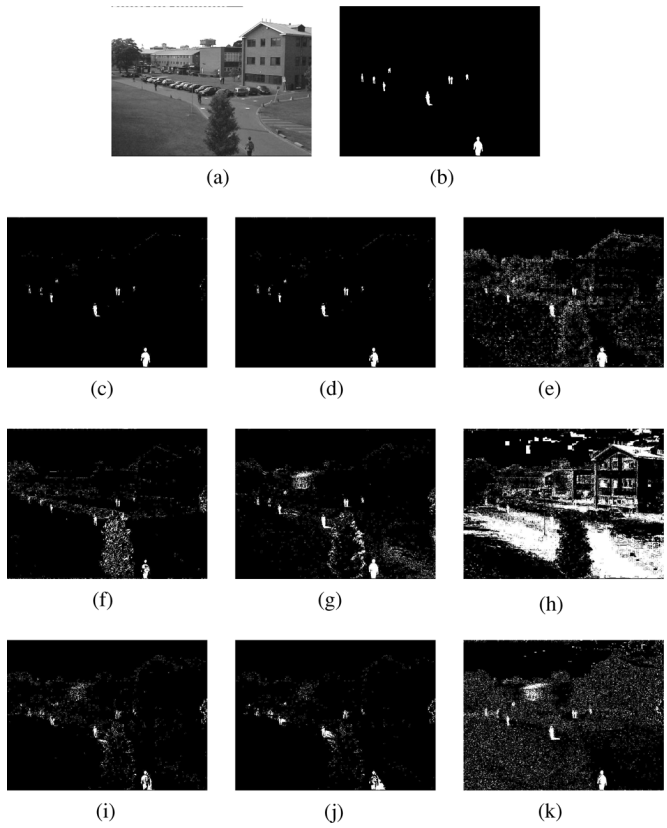


Fig. 6. Comparative background/foreground segmentation maps of nine background subtraction techniques for one frame taken from the "pets" sequence. Here too, the segmentation maps of ViBe are the closest to the ground-truth reference. (a) Input image. (b) Ground-truth. (c) ViBe (RGB). (d) ViBe (gray). (e) Bayesian histogram. (f) Codebook. (g) EGMM [Zivkovic]. (h) GMM [Li *et al.*]. (i) Gaussian model. (j) First order filter. (k) Sigma-Delta Z.

### *B. Comparison With Other Techniques*

We now compare the results of ViBe with those of five state-of-the-art background subtraction algorithms and two basic methods: 1) the Gaussian mixture model proposed in [47] (hereafter referred to as GMM); 2) the Gaussian mixture model of [50] (referred to as EGMM); 3) the Bayesian algorithm based upon histograms introduced in [70]; 4) the codebook algorithm [62]; 5) the zipfian $\Sigma$-$\Delta$ estimator of [37]; 6) a single Gaussian model with an adaptive variance (named "Gaussian model" hereafter); and 7) the first-order low-pass filter (that is $B_t = \alpha I_t + (1-\alpha)B_{t-1}$, where $I_t$ and $B_t$ are respectively the input and background images at time $t$), which is used as a baseline.

The first-order low-pass filter was tested using a fading factor $\alpha$ of 0.05 and a detection threshold $T$ of 20. A similar fading factor $\alpha$ of 0.05 was used for the Gaussian model. The GMM of [70] and the Bayesian algorithm of [47] were tested using their implementations available in Intel's IPP image processing library. For the EGMM algorithm of [50], we used the implementation available on the author's website[1]. The authors of the zipfian $\Sigma$-$\Delta$ filter were kind enough to provide us with their code to test their method. We implemented the codebook algorithm ourselves and used the following parameters: 50 training frames, $\lambda = 34$, $\epsilon_1 = 0.2$, $\epsilon_2 = 50$, $\alpha = 0.4$, and $\beta = 1.2$.

[1]http://staff.science.uva.nl/~zivkovic/DOWNLOAD.html

ViBe was tested with the default values proposed in this paper: $N = 20, R = 20, \sharp_{\min} = 2$, and $\phi = 16$. Most of the algorithms were tested using the RGB color space; the codebook uses its own color space, and the $\Sigma - \Delta$ filter implementation works on grayscale image. In addition, we implemented a grayscale version of ViBe.

Figs. 5 and 6 show examples of foreground detection for one typical frame of each sequence. Foreground and background pixels are shown in white and black respectively.

Visually, the results of ViBe look better and are the closest to ground-truth references. This is confirmed by the PCC scores; the PCC scores of the nine comparison algorithms for both sequences are shown in Fig. 7.

We also compared the computation times of these nine algorithms with a profiling tool, and expressed the computation times in terms of achievable framerates. Fig. 8 shows their average processing speed on our platform (2.67 GHz Core i7 CPU, 6 GB of RAM, C implementation).

We did not optimize the code of the algorithms explicitly, except in the case of the $\Sigma - \Delta$ algorithm, which was optimized by its authors, the algorithms of the IPP library (GMM and Bayesian histogram), which are optimized for Intel processors, and ViBe to some extent. To speed up operations involving random numbers in ViBe, we used a buffer prefilled with random numbers.

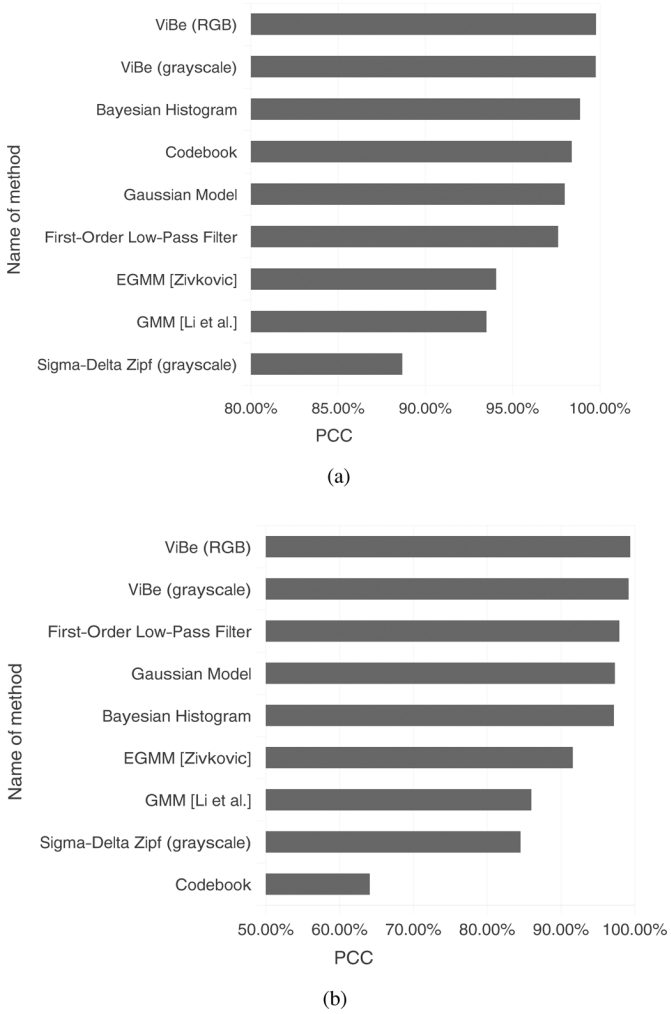


Fig. 7. PCCs of nine background subtraction techniques. ViBe has the highest PCCs. (a) Results for the first sequence ("house"). (b) Results for the second sequence ("pets").

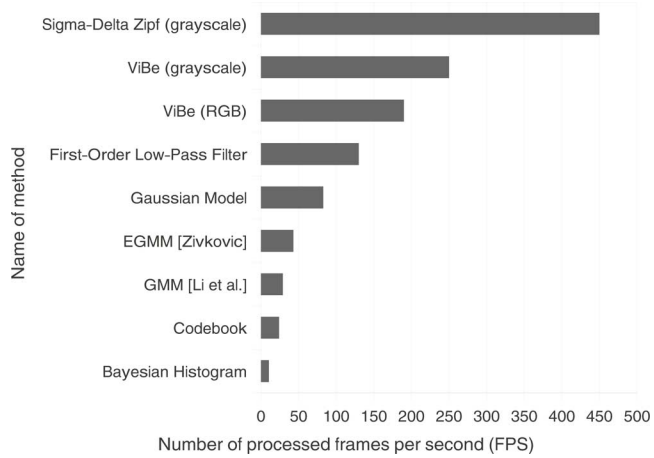


Fig. 8. Processing speed, expressed in terms of FPS, of nine background subtraction techniques for $640 \times 480$ pixels wide images.

We see that ViBe clearly outperforms the seven other techniques: its PCCs are the highest for both sequences and its processing speed is as high as a framerate of 200 FPS, that is five times more than algorithms optimized by Intel. Compare these figures to those obtained by the algorithm proposed by Chiu *et al.* [71] recently; they claim to segment $320 \times 240$ images at a framerate of around 40 FPS. A simple rescaling to the size of our images lowers this value to 10 FPS.

The only method faster than ViBe is the zipfian $\Sigma$-$\Delta$ estimator, whose PCC is 12 to 15% smaller than that of ViBe. The authors of the zipfian sigma-delta algorithm provided us with postprocessed segmentation maps of the "house" sequence which exhibit an improved PCC but a the cost of a lower processing speed. One can wonder how it is possible that ViBe runs faster than simpler techniques such as the first-order filter model. We discuss this question in Appendix B.

In terms of PCC scores, only the Bayesian algorithm of [70] based upon histograms competes with ViBe. However, it is more than 20 times slower than ViBe. As shown in Figs. 5 and 6, the grayscale and the color versions of ViBe manage to combine both a very small rate of FP and a sharp detection of the foreground pixels. The low FP rate of ViBe eliminates the need for any postprocessing, which further alleviates the total computational cost of the foreground detection.

Next, we concentrate on the specific strengths of ViBe: fast ghost suppression, intrinsic resilience to camera shake, noise resilience, and suitability for embedding.

### C. Faster Ghost Suppression

A background model has to adapt to modifications of the background caused by changing lighting conditions but also to those caused by the addition, removal, or displacement of some of its parts. These events are the principal cause of the appearance of ghosts: regions of connected foreground points that do not correspond to any real object.

When using a detection support map or a related technique to detect and suppress ghosts, it is very hard, if not impossible, to distinguish ghosts from foreground objects that are currently static. As a result, real foreground objects are included in the background model if they remain static for too long. This is a correct behavior since a static foreground object must eventually become part of the background after a given time. It would be better if ghosts were included in the background model more rapidly than real objects, but this is impossible since they cannot be distinguished using a detection support map.

Our spatial update mechanism speeds up the inclusion of ghosts in the background model so that the process is faster than the inclusion of real static foreground objects. This can be achieved because the borders of the foreground objects often exhibit colors that differ noticeably from those of the samples stored in the surrounding background pixel models. When a foreground object stops moving, the information propagation technique described in Section III-C.4 updates the pixel models located at its borders with samples coming from surrounding background pixels. But these samples are irrelevant: their colors do not match at all those of the borders of the object. In subsequent frames, the object remains in the foreground, since background samples cannot diffuse inside the foreground object via its borders.

By contrast, a ghost area often shares similar colors with the surrounding background. When background samples from the area surrounding the ghost try to diffuse inside the ghost, they are

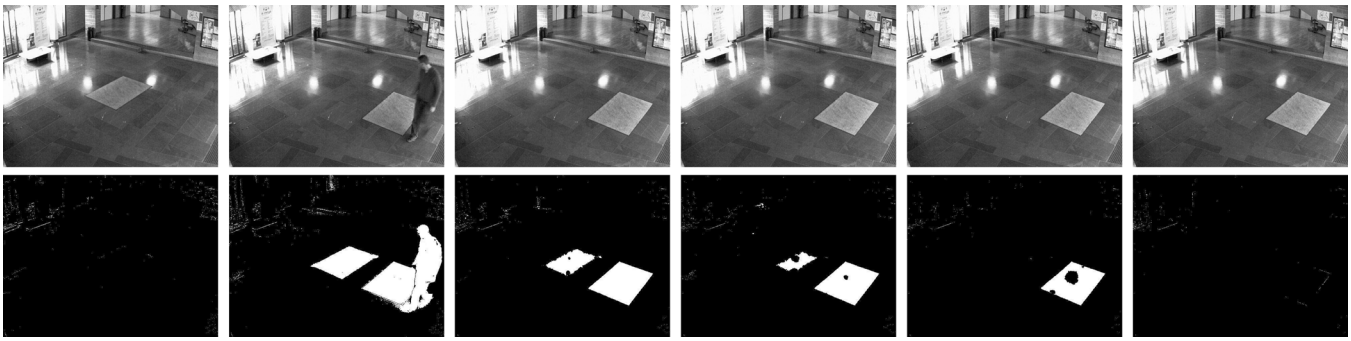

Fig. 9. Fast suppression of a ghost. In this scene, an object (a carpet) is moved, leaving a ghost behind it in the background, and is detected as being part of the foreground. It can be seen that the ghost is absorbed into the background model much faster than the foreground region corresponding to the real physical object.

likely to match the actual color of the image at the locations where they are diffused. As a result, the ghost is progressively eroded until it disappears entirely. Fig. 9 illustrates this discussion.

The speed of this process depends upon the texture of the background: the faster ghost suppressions are obtained with backgrounds void of edges. Furthermore, if the color of the removed object is close to that of the uncovered background area, the absorption of the ghost is faster. When needed, the speed of the ghost suppression process can be tuned by adapting the time subsampling factor $\phi$. For example, in the sequence displayed in Fig. 9, if we assume a framerate of 30 FPS, the ghost fades out after 2 s for a time subsampling factor $\phi$ equal to 1. However, if we set $\phi$ to 64, it takes 2 min for ViBe to suppress the ghost completely. For the sake of comparison, the Bayesian histogram algorithm suppresses the same ghost area in 5 s.

One may ask how static foreground objects will ultimately be included in the background model. The responsibility for the absorption of foreground pixels into the background lies with the noise inevitably present in the video sequence. Due to the noise, some pixels of the foreground object end up in the background, and then serve as background seeds. Consequently, their models are corrupted with foreground samples. These samples later diffuse into their neighboring models, as a result of the spatial propagation mechanism of the background samples, and allow a slow inclusion of foreground objects in the background.

### *D. Resistance to Camera Displacements*

In many situations, small displacements of the camera are encountered. These small displacements are typically due to vibrations or wind and, with many other techniques, they cause significant numbers of false foreground detections.

Another obvious benefit of the spatial consistency of our background model is an increased robustness against such small camera movements (see Fig. 10). Since samples are shared between neighboring pixel models, small displacements of the camera introduce very few erroneous foreground detections.

ViBe also has the capability of dealing with large displacements of the camera, at the price of a modification of the base algorithm. Since our model is purely pixel-based, we can make it able to handle moving cameras by allowing pixel models to follow the corresponding physical pixels according to the movements of the camera. The movements of the camera can be estimated either using embedded motion sensors or directly from the video stream using an algorithmic technique. This concept is illustrated in Figs. 11 and 12. The first series shows images taken from an old DARPA challenge. The camera pans the scene from left to right and the objective is to follow the car. Fig. 12 shows a similar scenario acquired with a Pan-Tilt Zoom video-surveillance camera; the aim here is to track the person.

Fig. 10. Background/foreground segmentation maps for a slightly moving camera. If spatial propagation is deactivated, the camera motions produce false positives in high-frequency areas (image in the center), while the activation of spatial propagation avoids a significant proportion of false positives (right-hand image).

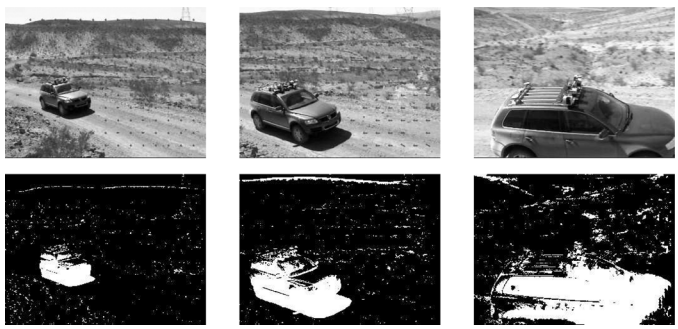

Fig. 11. Background/foreground segmentation maps for a sequence taken with a moving camera (from the DARPA challenge)

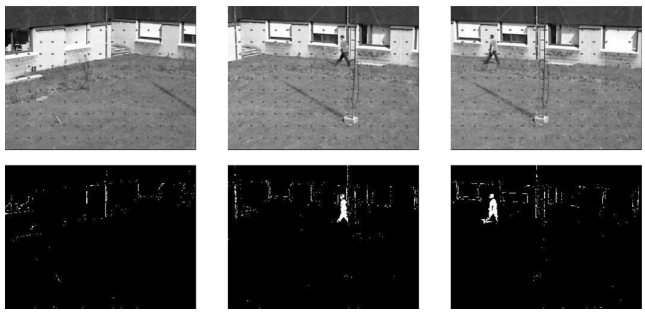

Fig. 12. Background/foreground segmentation maps for a sequence taken with a moving camera (surveillance camera).

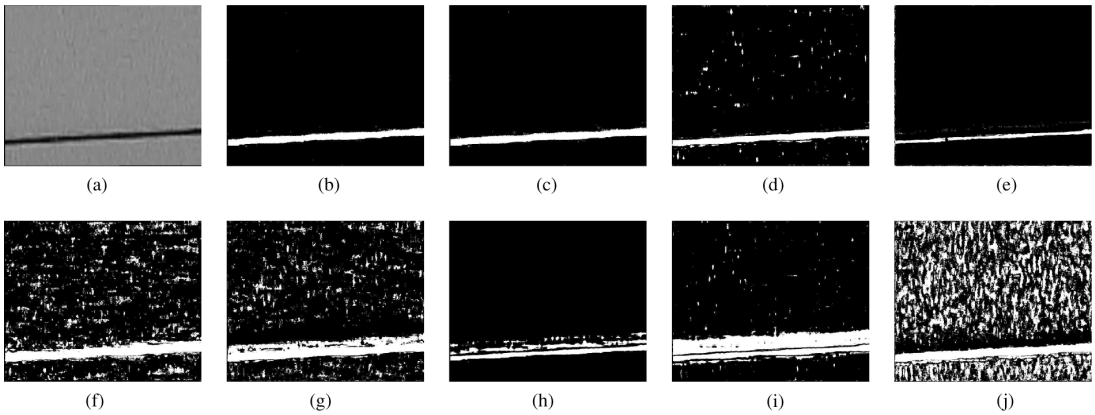


Fig. 13. Background/foreground segmentation maps for one frame taken from the noisy "cable" sequence. (a) Input image. (b) ViBe (RGB). (c) ViBe (grayscale). (d) GMM [Li *et al.*]. (e) Codebook. (f) Bayesian histogram. (g) EGMM [Zivkovic]. (h) Gaussian model. (i) First order filter. (j) Sigma-Delta Zipf.

To produce the images of Figs. 11 and 12, the displacement vector between two consecutive frames is estimated for a subset of background points located on a regularly spaced grid using Lucas and Kanade's optical flow estimator [72]. The global displacement vector of the camera is computed by averaging these pixel-wise displacement vectors. The pixel models are then relocated according to the displacement of the camera inside a larger mosaic reference image. The background model of pixels that correspond to areas seen for the first time is initialized instantaneously using the technique described in Section III-B. It can be seen that, even with such a simple technique, the results displayed in Figs. 11 and 12 are promising.

### *E. Resilience to Noise*

To demonstrate the resilience of ViBe to noise, we compared it to seven other techniques on a difficult noisy sequence (called "cable"). This sequence shows an oscillating electrical cable filmed at a large distance with a $40\times$ optical zoom. As can be seen in Fig. 13(a), the difficult acquisition conditions result in a significant level of noise in the pixel values. Background/foreground segmentation maps displayed in Fig. 13 demonstrate that ViBe is the only technique that manages to combine a low rate of FP with both a precise and accurate detection of the foreground pixels.

Two factors must be credited for ViBe's high resilience to noise. The first originates from our design, allowing the pixel models of ViBe to comprise *exclusively* observed pixel values. The pixel models of ViBe adapt to noise automatically, as they are constructed from noisy pixel values. The second factor is the pure conservative update scheme used by ViBe (see Section III-C). By relying on pixel values classified exclusively as background, the model update policy of ViBe prevents the inclusion of any outlier in the pixel models. As a result, these two factors ensure a continuous adaptation to the noise present in the video sequence while maintaining coherent pixel models.

### *F. Downscaled Version and Embedded Implementation*

Since ViBe has a low computational cost (see Fig. 8) and relies exclusively on integer computations, it is particularly well suited to an embedded implementation. Furthermore, the computational cost of ViBe can be further reduced by using low values for $N$ and $\sharp_{\min}$. Appendix B provides some implementation details and compares the complexity of ViBe with respect to the complexity of the first-order filter model.

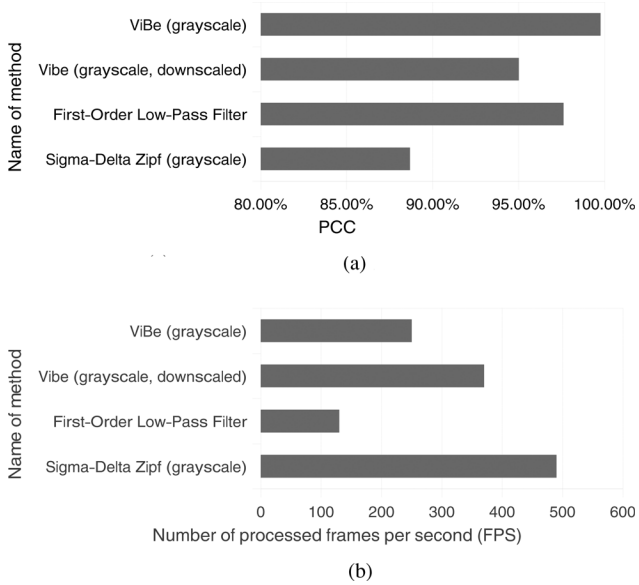


Fig. 14. PCCs and processing speeds of fast techniques, expressed in FPS, including a downscaled version of ViBe which requires only one comparison and one byte of memory per pixel. (a) PCC. (b) FPS for images of $640 \times 480$ pixels.

In Fig. 14, we give the PCC scores and framerates for a downscaled version of ViBe, which uses the absolute minimum of one comparison and one byte of memory per pixel. We also give the PCC scores and framerates for the full version of ViBe and for the two faster techniques from our tests in Section IV-B. One can see, on the left hand side of the graph in Fig. 14, that the downscaled version of ViBe maintains a high PCC. Note that its PCC is higher than that of the two GMM-based techniques tested in Section IV-B [see Fig. 7(a)]. In terms of processing speed or framerate, the zipfian $\Sigma$-$\Delta$ filter method of [37] is the only one to be faster than the downscaled version of ViBe. However, a postprocessing step of the segmentation map is necessary to increase the low PCC score of the zipfian $\Sigma$-$\Delta$ method, and the computational cost induced by this postprocessing process reduces the framerate significantly.


Fig. 15. Embedded implementation of ViBe in a Canon camera.

To illustrate the low computational cost of ViBe and its simplicity, we embedded our algorithm in a digital camera. The porting work of ViBe on a *Canon PowerShot SD870 IS* was performed with a modified version of the open source alternative firmware CHDK[2]. Parameters of ViBe were set to $N = 5$ and $\sharp_{\min} = 1$. Despite the camera's low speed ARM processor, we managed to process six frames of $320 \times 240$ pixels wide images per second on average. The result is shown in Fig. 15.

## V. Conclusion

In this paper, we introduced a universal sample-based background subtraction algorithm, called ViBe, which combines three innovative techniques.

First, we proposed a classification model that is based upon a small number of correspondences between a candidate value and the corresponding background pixel model. Second, we explained how ViBe can be initialized with a single frame. This frees us from the need to wait for several seconds to initialize the background model, an advantage for image processing solutions embedded in digital cameras and for short sequences. Finally, we presented our last innovation: an original update mechanism. Instead of keeping samples in the pixel models for a fixed amount of time, we ignore the insertion time of a pixel in the model and select a value to be replaced randomly. This results in a smooth decaying lifespan for the pixel samples, and enables an appropriate behavior of the technique for wider ranges of background evolution rates while reducing the required number of samples needing to be stored for each pixel model. Furthermore, we also ensure the spatial consistency of the background model by allowing samples to diffuse between neighboring pixel models. We observe that the spatial process is responsible for a better resilience to camera motions, but that it also frees us from the need to postprocess segmentation maps in order to obtain spatially coherent results. To be effective, the spatial propagation technique and update mechanism are combined with a strictly conservative update scheme: no foreground pixel value should ever be included in any background model.

After a description of our algorithm, we determined optimal values for all the parameters of the method. Using this set of parameter values, we then compared the classification scores and processing speeds of ViBe with those of seven other background subtraction algorithms on two sequences. ViBe is shown to outperform all of these algorithms while being faster than six of them. Finally, we discussed the performance of a downscaled version of ViBe, which can process more than 350 FPS on our platform. This downscaled version was embedded in a digital camera to prove its suitability for low speed platforms. Interestingly, we found that a version of ViBe downscaled to the absolute minimum amount of resources for any background subtraction algorithm (i.e., one byte of memory and one comparison with a memorized value per pixel) performed better than the state-of-the-art algorithms in terms of the PCC criterion. ViBe might well be a new milestone for the large family of background subtraction algorithms.

Please note that programs and object-code are available at http://www.motiondetection.org.

[2]http://chdk.wikia.com

## Appendix A
## C-Like Source Code for Our Algorithm

Pseudo-code of ViBe for grayscale images, comprising default values for all the parameters of the method, is given hereafter.

```
// fixed parameters for ViBe
// number of samples per pixel
int N = 20;
// radius of the sphere
int R = 20;
// number of close samples for being
// part of the background (bg)
int #_min = 2;
// amount of random subsampling
int phi = 16;
// data
int width, height;
// current image
byte image[width][height];
// background model
byte samples[width][height][N];
// background/foreground segmentation map
byte segMap[width][height];
// background and foreground identifiers
byte background = 0;
byte foreground = 255;
// for each pixel
for (int x = 0; x < width; x++){
    for (int y = 0; y < height; y++){
        // 1. Compare pixel to background model
        int count = 0, index = 0, dist = 0;
```

```
while ((count < ♯min)&&(index < N)){
    // Euclidean distance computation
    dist = EuclidDist(image[x][y],
        samples[x][y][index]);
    if (dist < R){
        count++;
    }
    index++;
}
// 2. Classify pixel and update model
if (count >= ♯min){
        // store that image[x][y] ∈ background
        segMap[x][y] = background;
        // 3. Update current pixel model
        // get random number between 0 and φ − 1
        int rand = getRandomNumber(0, φ − 1);
        if (rand == 0){ // random subsampling
            // replace randomly chosen sample
            rand = getRandomNumber(0, N − 1);
            samples[x][y][rand] = image[x][y];
        }
        // 4. Update neighboring pixel model
        rand = getRandomNumber(0, φ − 1);
        if (rand == 0){ // random subsampling
            // choose neighboring pixel randomly
            int x_NG, y_NG;
            x_NG = getRandomNeighbrXCoordinate(x);
            y_NG = getRandomNeighbrYCoordinate(y);
            // replace randomly chosen sample
            rand = getRandomNumber(0, N − 1);
            samples[x_NG][y_NG][rand] = image[x][y];
        }
    }
    else{//count < ♯min
        // store that image[x][y] ∈ foreground
        segMap[x][y] = foreground;
    }
  }
}
```

## APPENDIX B
## IMPLEMENTATION DETAILS, AND COMPLEXITY ANALYSIS OF VIBE AND THE FIRST-ORDER MODEL

As computation times of hardware or software operations might depend upon the processor or the compiler, it is hard to provide an exact analysis of the computation times. Instead, we present the steps involved for the computation of ViBe and the first-order filter model and evaluate the number of operations involved.

For ViBe, the evaluation runs as follows:

- Segmentation step:
  Remember that we compare a new pixel value to background samples to find two matches ($\sharp_{\min} = 2$). Once two matches have been found, we step over to the next pixel and ignore the remaining background samples. Operations involved during the segmentation step are:
  — comparison of the current pixel value with the values of the background model. Most of the time, the two first values of the background model of a pixel are close to the new pixel value. Therefore, we consider 2,5 (byte) comparisons on average per pixel (this is an experimentally estimated value).
  — 1,5 (byte) comparisons of the counter to check if there are at least two matching values in the model; we only need to compare the counter value after the comparison between the current pixel value and the second value of the background model.
- Update step:
  — 1 pixel substitution per 16 background pixels (the update factor, $\phi$, is equal to 16). Because we have to choose the value to substitute and access the appropriate memory block in the model, we perform an addition on memory addresses. Then we perform a similar operation, for a pixel in the neighborhood (first we locate which pixel in the neighborhood to select, then which value to substitute).
  In total, we evaluate the cost of the update step as three additions on memory addresses per 16 background pixels.
- Summary (average per pixel, assuming that most pixels belong to the background):
  — 4 subtractions on bytes.
  — $(3)/(16)$ addition on memory addresses.

For the first-order model, we have:

- Segmentation step:
  — 1 pixel comparison between an integer and a double number.
- Update step:
  — two multiplications and one addition on doubles, to perform $B_t = \alpha I_t + (1 - \alpha)B_{t-1}$.
- Summary (per pixel):
  — two multiplications and two additions on doubles

From this comparison, it appears that, once the random numbers are precalculated, the number of operations for ViBe is similar to that of the first-order filter model. However, if processors deal with "integer" (single byte) numbers faster than "double" numbers or if an addition is computed in less time than a multiplication, ViBe is faster than the first-order model.

## ACKNOWLEDGMENT

The authors would like to thank A. Manzanera, who provided the code for his algorithms, and to Z. Zivkovic for publishing his code on the Internet.

## REFERENCES

[1] O. Barnich and M. Van Droogenbroeck, "ViBe: A powerful random technique to estimate the background in video sequences," in *Proc. Int. Conf. Acoust., Speech Signal Process.*, Apr. 2009, pp. 945–948.

[2] M. Van Droogenbroeck and O. Barnich, Visual Background Extractor p. 36, Jan. 2009, World Intellectual Property Organization, WO 2009/007198.

[3] A. McIvor, "Background subtraction techniques," in *Proc. Image Vis. Comput.*, Auckland, New Zealand, Nov. 2000.

[4] R. Radke, S. Andra, O. Al-Kofahi, and B. Roysam, "Image change detection algorithms: A systematic survey," *IEEE Trans. Image Process.*, vol. 14, pp. 294–307, Mar. 2005.

[5] Y. Benezeth, P. Jodoin, B. Emile, H. Laurent, and C. Rosenberger, "Review and evaluation of commonly-implemented background subtraction algorithms," in *Proc. IEEE Int. Conf. Pattern Recognit.*, Dec. 2008, pp. 1–4.

[6] S. Elhabian, K. El-Sayed, and S. Ahmed, "Moving object detection in spatial domain using background removal techniques—State-of-art," *Recent Pat. Comput. Sci.*, vol. 1, pp. 32–54, Jan. 2008.

[7] M. Piccardi, "Background subtraction techniques: A review," in *Proc. IEEE Int. Conf. Syst., Man Cybern.*, The Hague, The Netherlands, Oct. 2004, vol. 4, pp. 3099–3104.

[8] D. Parks and S. Fels, "Evaluation of background subtraction algorithms with post-processing," in *Proc. IEEE Int. Conf. Adv. Video Signal Based Surveillance*, Santa Fe, New Mexico, Sep. 2008, pp. 192–199.

[9] T. Bouwmans, F. El Baf, and B. Vachon, "Statistical background modeling for foreground detection: A survey," in *Handbook of Pattern Recognition and Computer Vision (Volume 4)*. Singapore: World Scientific, Jan. 2010, ch. 3, pp. 181–199.

[10] M. Seki, T. Wada, H. Fujiwara, and K. Sumi, "Background subtraction based on cooccurrence of image variations," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Los Alamitos, CA, Jun. 2003, vol. 2, pp. 65–72.

[11] P. Power and J. Schoonees, "Understanding background mixture models for foreground segmentation," in *Proc. Image Vis. Comput.*, Auckland, New Zealand, Nov. 2002, pp. 267–271.

[12] N. Oliver, B. Rosario, and A. Pentland, "A Bayesian computer vision system for modeling human interactions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 831–843, Aug. 2000.

[13] D.-M. Tsai and S.-C. Lai, "Independent component analysis-based background subtraction for indoor surveillance," *IEEE Trans. Image Process.*, vol. 18, no. 1, pp. 158–167, Jan. 2009.

[14] H.-H. Lin, T.-L. Liu, and J.-C. Chuang, "Learning a scene background model via classification," *IEEE Signal Process. Mag.*, vol. 57, no. 5, pp. 1641–1654, May 2009.

[15] L. Maddalena and A. Petrosino, "A self-organizing approach to background subtraction for visual surveillance applications," *IEEE Trans. Image Process.*, vol. 17, no. 7, pp. 1168–1177, Jul. 2008.

[16] V. Cevher, A. Sankaranarayanan, M. Duarte, D. Reddy, R. Baraniuk, and R. Chellappa, "Compressive sensing for background subtraction," in *Proc. Eur. Conf. Comput. Vis.*, Oct. 2008, pp. 155–168.

[17] M. Dikmen and T. Huang, "Robust estimation of foreground in surveillance videos by sparse error estimation," in *Proc. IEEE Int. Conf. Pattern Recognit.*, Tampa, FL, Dec. 2008, pp. 1–4.

[18] S. Cohen, "Background estimation as a labeling problem," in *Proc. Int. Conf. Comput. Vis.*, Beijing, China, Oct. 2005, vol. 2, pp. 1034–1041.

[19] V. Mahadevan and N. Vasconcelos, "Spatiotemporal saliency in dynamic scenes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 1, pp. 171–177, Jan. 2010.

[20] M. Sivabalakrishnan and D. Manjula, "An efficient foreground detection algorithm for visual surveillance system," *Int. J. Comput. Sci. Network Sec.*, vol. 9, pp. 221–227, May 2009.

[21] A. Cavallaro and T. Ebrahimi, "Video object extraction based on adaptive background and statistical change detection," in *Proc. Vis. Commun. Image Process.*, Jan. 2001, pp. 465–475.

[22] A. El Maadi and X. Maldague, "Outdoor infrared video surveillance: A novel dynamic technique for the subtraction of a changing background of IR images," *Infrared Phys. Technol.*, vol. 49, pp. 261–265, Jan. 2007.

[23] R. Abbott and L. Williams, "Multiple target tracking with lazy background subtraction and connected components analysis," *Mach. Vis. Appl.*, vol. 20, pp. 93–101, Feb. 2009.

[24] B. Shoushtarian and H. Bez, "A practical adaptive approach for dynamic background subtraction using an invariant colour model and object tracking," *Pattern Recognit. Lett.*, vol. 26, pp. 5–26, Jan. 2005.

[25] J. Cezar, C. Rosito, and S. Musse, "A background subtraction model adapted to illumination changes," in *Proc. IEEE Int. Conf. Image Process.*, Oct. 2006, pp. 1817–1820.

[26] J. Davis and V. Sharma, "Robust background-subtraction for person detection in thermal imagery," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Washington, DC, Jun. 2004, vol. 8, p. 128.

[27] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland, "Pfinder: Real-time tracking of the human body," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 780–785, Jul. 1997.

[28] K. Toyama, J. Krumm, B. Brumitt, and M. Meyers, "Wallflower: Principles and practice of background maintenance," in *Proc. Int. Conf. Comput. Vis.*, Kerkyra, Greece, Sep. 1999, pp. 255–261.

[29] D. Koller, J. Weber, and J. Malik, "Robust multiple car tracking with occlusion reasoning," in *Proc. Eur. Conf. Comput. Vis.*, Stockholm, Sweden, May 1994, pp. 189–196.

[30] J. Davis and V. Sharma, "Background-subtraction in thermal imagery using contour saliency," *Int. J. Comput. Vis.*, vol. 71, pp. 161–181, Feb. 2007.

[31] C. Jung, "Efficient background subtraction and shadow removal for monochromatic video sequences," *IEEE Trans. Multimedia*, vol. 11, no. 3, pp. 571–577, Apr. 2009.

[32] D. Gutchess, M. Trajkovics, E. Cohen-Solal, D. Lyons, and A. Jain, "A background model initialization algorithm for video surveillance," in *Proc. Int. Conf. Comput. Vis.*, Vancouver, BC, Jul. 2001, vol. 1, pp. 733–740.

[33] I. Haritaoglu, D. Harwood, and L. Davis, "$W^4$: Real-time surveillance of people and their activities," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 809–830, Aug. 2000.

[34] J. Jacques, C. Jung, and S. Musse, "Background subtraction and shadow detection in grayscale video sequences," in *Proc. Brazilian Symp. Comput. Graph. Image Process.*, Natal, Brazil, Oct. 2005, pp. 189–196.

[35] A. Manzanera and J. Richefeu, "A robust and computationally efficient motion detection algorithm based on sigma-delta background estimation," in *Proc. Indian Conf. Comput. Vis., Graph. Image Process.*, Kolkata, India, Dec. 2004, pp. 46–51.

[36] A. Manzanera and J. Richefeu, "A new motion detection algorithm based on $\Sigma - \Delta$ background estimation," *Pattern Recognit. Lett.*, vol. 28, pp. 320–328, Feb. 2007.

[37] A. Manzanera, "$\Sigma - \Delta$ background subtraction and the Zipf law," in *Proc. Progr. Pattern Recognit., Image Anal. Appl.*, Nov. 2007, pp. 42–51.

[38] L. Lacassagne, A. Manzanera, J. Denoulet, and A. Mérigot, "High performance motion detection: Some trends toward new embedded architectures for vision systems," *J. Real-Time Image Process.*, vol. 4, pp. 127–146, Jun. 2009.

[39] L. Lacassagne, A. Manzanera, and A. Dupret, "Motion detection: Fast and robust algorithms for embedded systems," in *Proc. IEEE Int. Conf. Image Process.*, Cairo, Egypt, Nov. 2009, pp. 3265–3268.

[40] C. Stauffer and E. Grimson, "Adaptive background mixture models for real-time tracking," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Ft. Collins, CO, Jun. 1999, vol. 2, pp. 246–252.

[41] C. Stauffer and E. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 747–757, Aug. 2000.

[42] B. Lei and L. Xu, "Real-time outdoor video surveillance with robust foreground extraction and object tracking via multi-state transition management," *Pattern Recognit. Lett.*, vol. 27, pp. 1816–1825, Nov. 2006.

[43] Y. Wang, T. Tan, K. Loe, and J. Wu, "A probabilistic approach for foreground and shadow segmentation in monocular image sequences," *Pattern Recognit.*, vol. 38, pp. 1937–1946, Nov. 2005.

[44] Y. Wang, K. Loe, and J. Wu, "A dynamic conditional random field model for foreground and shadow segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 2, pp. 279–289, Feb. 2006.

[45] O. Barnich, S. Jodogne, and M. Van Droogenbroeck, "Robust analysis of silhouettes by morphological size distributions," in *Advanced Concepts for Intelligent Vision Systems (ACIVS 2006), Vol. 4179 of Lecture Notes on Computer Science*. New York: Springer-Verlag, Sep. 2006, pp. 734–745.

[46] J.-S. Hu and T.-M. Su, "Robust background subtraction with shadow and highlight removal for indoor surveillance," *EURASIP J. Appl. Signal Process.*, vol. 2007, pp. 108–108, Jan. 2007.
[47] P. KaewTraKulPong and R. Bowden, "An improved adaptive background mixture model for real-time tracking with shadow detection," in *Proc. Eur. Workshop Adv. Video Based Surveillance Syst.*, London, U.K., Sep. 2001.
[48] D. Lee, "Effective Gaussian mixture learning for video background subtraction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 5, pp. 827–832, May 2005.
[49] Q. Zang and R. Klette, "Robust background subtraction and maintenance," in *Proc. IEEE Int. Conf. Pattern Recognit.*, Washington, DC, Aug. 2004, vol. 2, pp. 90–93.
[50] Z. Zivkovic, "Improved adaptive gausian mixture model for background subtraction," in *Proc. IEEE Int. Conf. Pattern Recognit.*, Cambridge, U.K., Aug. 2004, vol. 2, pp. 28–31.
[51] B. White and M. Shah, "Automatically tuning background subtraction parameters using particle swarm optimization," in *Proc. IEEE Int. Conf. Multimedia Expo*, Beijing, China, Jul. 2007, pp. 1826–1829.
[52] P. Varcheie, M. Sills-Lavoie, and G.-A. Bilodeau, "A multiscale region-based motion detection and background subtraction algorithm," *Sensors*, vol. 10, pp. 1041–1061, Jan. 2010.
[53] A. Elgammal, D. Harwood, and L. Davis, "Non-parametric model for background subtraction," in *Proc. 6th Eur. Conf. Comput. Vis.*, London, U.K., Jun.–Jul. 2000, pp. 751–767.
[54] A. Srivastava, A. Lee, E. Simoncelli, and S.-C. Zhu, "On advances in statistical modeling of natural images," *J. Math. Imag. Vis.*, vol. 18, pp. 17–33, Jan. 2003.
[55] A. Elgammal, R. Duraiswami, D. Harwood, and L. Davis, "Background and foreground modeling using nonparametric kernel density estimation for visual surveillance," *Proc. IEEE*, vol. 90, no. 7, pp. 1151–1163, Jul. 2002.
[56] A. Mittal and N. Paragios, "Motion-based background subtraction using adaptive kernel density estimation," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Los Alamitos, CA, Jun.–Jul. 2004, vol. 2, pp. 302–309.
[57] Y. Sheikh and M. Shah, "Bayesian modeling of dynamic scenes for object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 11, pp. 1778–1792, Nov. 2005.
[58] Z. Zivkovic and F. van der Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction," *Pattern Recognit. Lett.*, vol. 27, pp. 773–780, May 2006.
[59] A. Tavakkoli, M. Nicolescu, G. Bebis, and M. Nicolescu, "Non-parametric statistical background modeling for efficient foreground region detection," *Mach. Vis. Appl.*, vol. 20, pp. 395–409, Oct. 2008.
[60] E. Monari and C. Pasqual, "Fusion of background estimation approaches for motion detection in non-static backgrounds," in *Proc. IEEE Int. Conf. Adv. Video Signal Based Surveillance*, London, U.K., Sep. 2007, pp. 347–352.
[61] K. Kim, T. Chalidabhongse, D. Harwood, and L. Davis, "Background modeling and subtraction by codebook construction," in *Proc. IEEE Int. Conf. Image Process.*, Singapore, Oct. 2004, vol. 5, pp. 3061–3064.
[62] K. Kim, T. Chalidabhongse, D. Harwood, and L. Davis, "Real-time foreground-background segmentation using codebook model," *Real-Time Imag.*, vol. 11, Special Issue on Video Object Processing, pp. 172–185, Jun. 2005.
[63] M. Wu and X. Peng, "Spatio-temporal context for codebook-based dynamic background subtraction," *Int. J. Electron. Commun.*, vol. 64, no. 8, pp. 739–747, 2010.
[64] H. Wang and D. Suter, "Background subtraction based on a robust consensus method," in *Proc. IEEE Int. Conf. Pattern Recognit.*, Washington, DC, Aug. 2006, pp. 223–226.
[65] H. Wang and D. Suter, "A consensus-based method for tracking: Modelling background scenario and foreground appearance," *Pattern Recognit.*, vol. 40, pp. 1091–1105, Mar. 2007.
[66] P.-M. Jodoin, M. Mignotte, and J. Konrad, "Statistical background subtraction using spatial cues," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 12, pp. 1758–1763, Dec. 2007.
[67] A. Criminisi, P. Pérez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Trans. Image Process.*, vol. 13, no. 9, pp. 1200–1212, Sep. 2004.
[68] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*. New York: McGraw-Hill, 1984.
[69] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, "Detecting moving objects, ghosts, and shadows in video streams," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 10, pp. 1337–1342, Oct. 2003.
[70] L. Li, W. Huang, I. Gu, and Q. Tian, "Foreground object detection from videos containing complex background," in *Proc. ACM Int. Conf. Multimedia*, Berkeley, CA, Nov. 2003, pp. 2–10.
[71] C.-C. Chiu, M.-Y. Ku, and L.-W. Liang, "A robust object segmentation system using a probability-based background extraction algorithm," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 4, pp. 518–528, Apr. 2010.
[72] B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. Int. Joint Conf. Artif. Intell.*, Vancouver, BC, Apr. 1981, pp. 674–679.

**Olivier Barnich** was born in Liège, Belgium. He received the electrical engineering degree and Ph.D. degree from the University of Liège, Belgium, in 2004 and 2010, respectively.

He is currently working for EVS Broadcast Equipment, Seraing, Belgium, a company that delivers production and playout systems for the broadcast of sports, news and entertainment. His research interests are mainly focused on motion detection, human detection and recognition, and machine learning enabled computer vision.

**Marc Van Droogenbroeck** (M'99) received the degree in electrical engineering and the Ph.D. degree from the University of Louvain (UCL, Belgium) in 1990 and 1994, respectively.

Since 1998, M. Van Droogenbroeck has been a member of the Faculty of Applied Sciences at the University of Liège, Belgium, where he is currently a Professor. During his Ph.D. he spent two years at the Center of Mathematical Morphology (CMM) of the School of Mines of Paris. In April 1994, he joined the New Development Department, Belgacom. He was appointed as the Head of the Belgian Delegation in the ISO/MPEG Committee and as a representative to the World Wide Web Consortium for two years. In 2003, he was a visiting Scientist at the CSIRO, Australia. His current interests are image processing, computer vision, mathematical morphology, fast algorithms, and video surveillance.