doi:10.3969/j.issn.1671-637X.2017.05.008

基于CUDA的并行视觉背景提取算法

田 毅[1], 汪克念[1], 郭冲宇[2], 赵长啸[1]
(1.中国民航大学天津市民用航空器适航与维修重点实验室,天津 300300;
2.西安邮电大学电子工程学院,西安 710061)

摘 要:为了提高视觉背景提取(ViBe)算法的检测效果和检测速度,对其进行改进和并行化处理。该算法基于像素及其邻域像素之间的差值构建最终的背景样本,减少边缘噪声对于检测精度的影响;背景样本更新阶段中使用自适应的时间二次抽样因子,加快消除单帧初始化所形成的"鬼影",提高算法的健壮性。同时分析了ViBe算法的并行点,并用CUDA语言实现该算法。实验结果表明,该算法能有效减少边缘噪声,快速消除"鬼影",且检测速度较CPU端有大幅提升。

关键词:目标检测;视觉背景提取;鬼影消除;自适应时间二次抽样因子
中图分类号:TP391 **文献标志码**:A **文章编号**:1671-637X(2017)05-0040-04

Parallel Visual Background Extractor Algorithm Based on CUDA

TIAN Yi[1], WANG Ke-nian[1], GUO Chong-yu[2], ZHAO Chang-xiao[1]
(1. Tianjin Key Laboratory of Civil Aircraft Airworthiness and Maintenance, Civil Aviation University of China, Tianjin 300300, China; 2. School of Electronic Engineering, Xi'an University of Posts and Telecommunications, Xi'an 710061, China)

Abstract: Improvements are made to ViBe (Visual Background extractor) algorithm for enhancing its test effect and test speed. The algorithm utilizes several pixel values from consecutive frames to train and construct the final samples of background during initialization, thus to decrease the ghost's effects on precision in detecting an object and partially remove the noises on the edge of background objects. In the process of pixel matching, a classified threshold and a match threshold for scene complexity of the pixel are adjusted dynamically to adapt complicated background scenes and diminish the disturbances of incorrect foreground pixels. In order to reinforce the robustness of the improved algorithm, an adaptive time-subsampling factor is utilized to accelerate elimination of the ghost caused by the departure of background objects and recovery of the background samples. Experiments show that the algorithm presented here can remove the ghost and noises effectively and is appropriate for complicated background scenes detections.

Key words: object detection; visual background extractor; ghost elimination; adaptive time-subsampling factor

0 引言

动目标检测是智能视频监控、行为识别等领域的核心问题[1],也是计算机视觉的重要研究方向。通常,运动目标检测算法可分为光流法[2]、帧间差分法[3]、背景差分法[4-6]3大类。其中:光流法计算量较大,不利于实时处理,并且容易受到光照的影响而产生误检;帧间差分法虽然计算量小、速度快,但是不易提取出完整的运动目标;背景差分法是目前较为常用的运动目标检测算法,其主要思想是为背景构建背景模型,通过比较当前帧与背景模型来提取出运动目标。视觉背景提取(Visual Background extractor,ViBe)算法是BARNICH O提出的一种非参数化背景差分算法[5-6],该算法对每个像素独立建模,具有初始化速度快、判断过程简单的特

收稿日期:2016-04-11 **修回日期**:2017-03-11
基金项目:民航联合研究基金(U1333120);民航科技创新引导资金重大专项(MHRD20140103);中国民航大学科研启动基金项目(2013QD04X)
作者简介:田 毅(1983—),男,陕西汉中人,硕士,助理研究员,研究方向为机载软、硬件设计与验证技术。

点，但也含有如下几个方面的缺点：一是对于高分辨率视频，由于其数据量较大，CPU 串行处理方式将耗费大量计算时间，不能应用于实时检测；二是随机选取像素邻域值初始化背景样本的方式会在场景中的物体边缘处检测出大量噪声点；三是通过一帧完成背景样本构建的方式在含有前景目标的情况下会导致“鬼影”的出现。因此，本文算法针对 ViBe 算法初始化、背景样本更新的不足进行改进，并分析 ViBe 算法的并行性，用 NIVIDIA CUDA 架构对提出的算法进行验证，极大地加速了算法的运行速度。实验结果表明，对于不同分辨率的检测视频，基于 CUDA 架构的 ViBe 算法检测速度有大幅提升。

1　视觉背景提取算法

ViBe 算法是一种快速的像素级前景检测算法，该算法为每个像素创建一个背景样本，采用随机选择策略初始化背景样本，通过对比当前像素与背景样本来判别该像素为前景或背景，并用随机更新和邻域扩散机制更新背景样本。算法核心过程如下所述。

1）背景样本的构建及其初始化。

为图像中的每个像素 x 创建一个背景样本 $M(x)$，即

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

式中：N 为背景样本大小，即背景样本中元素的个数；$v_i(1\leqslant i\leqslant N)$ 代表背景样本第 i 个值。

背景样本使用视频的第一帧图像进行初始化，任意一个像素 x 的 N 个背景样本元素值从其邻域像素中等概率随机选取，即

$$M(x)=\{v^1(y)\mid y\in N_G(x)\} \tag{2}$$

式中：$N_G(x)$ 为 x 的邻域像素集；y 为从像素 x 的邻域中随机选出的像素；$v^1(y)$ 为像素 y 的像素值。

2）前景判断。

假设像素 x 的像素值为 $v(x)$，定义 $S_R(v(x))$ 是以 $v(x)$ 为中心、R 为半径的圆。对于像素 x 的背景样本 $M(x)$，用 $\#\{S_R(v(x))\cap M(x)\}$ 表示 $M(x)$ 中落入 $S_R(v(x))$ 内背景样本元素的个数。

给定匹配阈值 #min，ViBe 算法需要通过比较 $\#\{S_R(v(x))\cap M(x)\}$ 和匹配阈值 #min 来确定像素 x 究竟是属于背景像素还是前景像素。如果像素 x 满足 $\#\{S_R(v(x))\cap M(x)\}\geqslant$#min，则将像素 x 判为背景；否则，判定为前景。

3）背景样本更新。

ViBe 算法采用随机更新和邻域扩散机制，对于被判定为背景的像素 x，有 $1/\varphi$（φ 为时间二次抽样因子）的概率用其当前像素值更新背景样本 $M(x)$，以及 $1/\varphi$ 的概率随机选择某个邻域像素 y，等概率随机选择 N 个背景样本元素中的一个更新。

2　改进的 ViBe 算法及其并行化实现

2.1　改进的 ViBe 算法

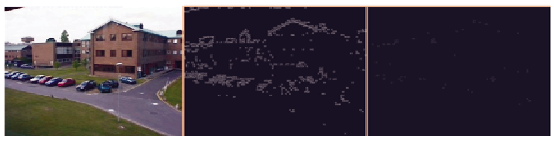
2.1.1　背景样本初始化

背景样本的构建对最终检测结果至关重要，ViBe 算法假设相邻的像素具有相似的时空分布，在视频序列第一帧使用一个像素的八邻域像素等概率随机填充该背景样本，这在大多数背景区域是有效的。但是，当背景像素处于物体边界处，这种假设将不成立，并导致并不具备相似性的像素被用于构建背景样本。进入下一帧后，由于一个像素与其背景样本不匹配，造成了物体边缘区域出现噪声点。

为此，对于任意像素 x，本文算法在视频序列第一帧依次用 x 及其八邻域内的像素 $y_i(1\leqslant i\leqslant 8)$ 的差值 d_i 按照式(3)标记其邻域像素，即

$$\tau_i=\begin{cases}0 & d_i>D\\ 1 & \text{其他}\end{cases} \tag{3}$$

式中：τ_i 为第 i 个邻域像素的标记；D 为固定阈值。在背景样本构建阶段，随机选取像素 x 及其八邻域内标记为 1 的像素构建其背景样本。本文算法取阈值 $D=20$，并将上述初始化方式与 ViBe 算法初始化方式进行对比，结果如图 1 所示。



a 输入帧　　b ViBe 算法初始化　c 改进算法初始化

图 1　ViBe 算法与改进算法初始化效果

Fig. 1　Initialization effect of ViBe algorithm and the improved algorithm

由图 1 的检测结果可看出，ViBe 算法的初始化方式在场景中物体的边缘处检测出了较多的噪声点，而本文算法改进的初始化方式则有效减少了边缘噪声干扰，仅凭肉眼已经难以识别噪声点，达到较为理想的检测效果。

2.1.2　背景样本更新

ViBe 算法利用第一帧图像信息构建背景样本，但当第一帧存在前景时，前景像素值被用于构建背景样本，从而在后续的检测过程中形成“鬼影”。虽然 ViBe 算法采用邻域扩散机制来逐步消除“鬼影”，但是这种机制并不能快速消除“鬼影”，这无疑会降低算法的精确性。“鬼影”产生的根本原因在于 ViBe 算法的初始化方式，当前景出现在视频图像第一帧时，ViBe 算法将前景目标覆盖区域内的像素初始化为背景样本。当前景离开这片区域后，真实的背景像素与其背景样本

不匹配,从而被误判为前景。文献[7-8]提出了前景点计数的方法来快速消除“鬼影”,即当像素被判定为前景点的次数大于某个阈值时便将其判定为背景,但这种做法易将运动缓慢或静止的前景误判为“鬼影”,从而影响检测结果的准确性。

ViBe 使用全局固定的时间二次抽样因子 φ,当像素 x 被判断为背景像素时,其有 $1/\varphi$ 的概率用其像素值更新邻域像素的背景样本,通过邻域扩散的方式逐渐消除“鬼影”,但其全局固定的时间二次抽样因子导致了“鬼影”将在很长的时间内存在于检测结果。本文算法基于 Σ-Δ 算法构建背景的思想,对每一个像素 $\{x_1,x_2,\cdots,x_k\}$ 建立与之对应的变量 $\{D_1,D_2,\cdots,D_k\}$,并初始化

$$D_1=D_2=\cdots=D_k=\cdots=0 \tag{4}$$

使用当前帧的检测结果与前一帧的检测结果作比较,通过比较结果调整 D_k 的值。

$$D_k=\begin{cases}D_k+D_{\text{inc}} & x_k^t\in\text{background}\ \ \&\&\ \ x_k^{t-1}\in\text{background}\\ D_k-D_{\text{dec}} & \text{其他}\end{cases} \tag{5}$$

式中,D_{inc},D_{dec}为固定值。设定阈值,时间二次抽样因子 φ 根据 D_k 的值自适应更新

$$\varphi=\begin{cases}\varphi_{\text{init}}-[(D_k-T)/10] & D_k>T\\ \varphi_{\text{init}}+[(T-D_k)/5] & \text{其他}\end{cases} \tag{6}$$

式中:φ_{init}为时间二次抽样因子初始值的取值;T 为固定阈值;[ ]为取整符号;D_k 的取值范围为 $D_k\in[0,120]$。

取上述参数 $\varphi_{\text{init}}=16$,$D_{\text{inc}}=1$,$D_{\text{dec}}=20$,阈值 $T=40$,与 ViBe 算法进行对比,实验结果见图 2。图 2 检测视频中第一帧出现前景,继而在检测结果中出现“鬼影”。

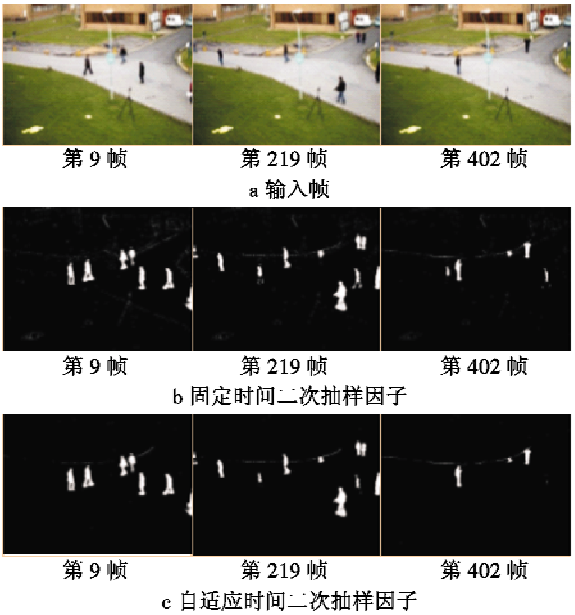


图 2 固定和自适应时间二次抽样因子检测结果

Fig. 2 Detection result of the fixed and the adaptive time-subsampling factor

由图 2 检测效果可以看出,本文算法消除“鬼影”的速度较快,同时有效减少了边缘噪声点,提高了算法的效率。

2.2 ViBe 算法的并行化分析及实现

ViBe 算法中,为了完成最终的检测,需要将像素的像素值与其对应的背景样本匹配,对于大小为 $W\times H$ 的视频,由 1 节可知,基于 CPU 端的串行 ViBe 算法时间复杂度为 $O(W\times H)$ 或 $O(n^2)$,这将使 CPU 端的 ViBe 算法无法实时处理高分辨率视频。

英伟达的 GPGPU 支持 CUDA 核,这些 CUDA 核中又包含了众多的线程,非常适合对数据密集型的应用进行加速。应用程序使用 GPU 加速分为 3 个阶段:1) CPU 端获取当前视频帧数据,并将数据由 RGB 空间转换到灰度空间;2) CPU 端将数据传送至 GPU 端,GPU 按照算法对数据进行并行处理;3) 将 GPU 处理后的数据传至 CPU,并生产最终的检测结果。其中,CPU 和 GPU 数据交换的大小会对 ViBe 算法的执行时间产生影响。

另外,根据阿姆达尔定律,算法的加速比并不取决于并行的部分,而是取决于不能并行处理的串行部分。因此,对算法的加速分成 2 个部分:1) 最大化算法可以并行的部分;2) 减少从 CPU 到 GPU 的数据交换。

通过分析 ViBe 算法,本文算法对每个像素进行独立建模,不同位置间的像素匹配过程彼此无关,其检测过程具有高度的并行性。因此,可以依据像素坐标给不同位置的像素分配独立的线程,基于 CUDA 架构实现算法在 GPU 端并行执行。

检测过程中线程数量在理想情况下为 $W\times H$(一般情况下 GPU 端的线程数可以满足需求),但在实际执行过程中,需要根据视频大小动态调整线程数量,具体表示为 *ViBe_kernel* < < < *grids*, *block* > > > (……)。其中,*block* 为二维数据,即 *block* = (x, y),x 表示线程块中每一行的线程数,y 表示线程块中每一列的线程数,本文算法设置为固定值 *block* = (32, 4);*grids* 为二维数据,即 *grids* = (x, y),x 表示线程格中每一行的线程块数量,y 表示线程格中每一列的线程块数量,其大小由视频图像大小决定。考虑到视频图像大小($W\times H$)并不一定是 *block.x* 与 *block.y* 的整数倍,所以本文算法使用向上取整的策略,根据图像大小($W\times H$)动态调整线程块的数量:*grids.x* = (W + *block.x* − 1)/ *block.x*, *grids.y* = (H + *block.y* − 1)/ *block.y*。

在 CUDA 实现该算法时,存储分配是非常重要的问题,如果分配不合理,会影响算法的加速比。本文算法所需内存包含 3 个部分:1) 每帧的图像数据;2) $N\times W\times H$ 大小的背景样本;3) 坐标偏移量。

考虑到视频图像在 CPU 端获取，并且检测过程中每一帧图像都需要由 CPU 端送至 GPU 端处理，并在检测结束后返还至 CPU 端，所以本文算法在视频图像第一帧使用函数 *cudaMalloc*( ) 在 GPU 端的 global memory 中开辟一块固定的内存区域用于存放每帧图像数据，并在对整个视频检测结束后使用函数 *cudaFree*( ) 释放这块内存区域；建立大小为 $N \times W \times H$ 的一维数组用于保存不同位置像素的背景样本，位于第 x 行、第 y 列的像素，其第 i 个背景样本元素位于一维数组 $i \times x \times W + y$ 处。由于 GPU 端需要使用像素与其背景样本进行比对，为了加快存取速度，本文算法同样在视频第一帧使用函数 *cudaMalloc*( ) 在 GPU 端开辟一块大小为 $N \times W \times H$ 的固定内存区域用于存放不同位置像素的 N 个背景样本元素；ViBe 算法中像素的背景样本由其自身及其八邻域像素共同维护更新，所以坐标偏移量的访问速度影响整个算法的检测速度。由于坐标偏移量在整个检测过程中为固定值，因此本文算法用 GPU 的常量内存存放坐标偏移量。

ViBe 算法对于样本更新采取的是随机的策略，所以随机数的产生是算法并行化的核心问题，为减少 CPU 与 GPU 之间的数据交换，在 GPU 端产生随机数，具体做法如下：1）GPU 端初始化随机数种子 *state*，其中，*state* 类型为 32 位整型；2）将 *state* 与固定值 T_1 相乘得到 s_1，并将 *state* 右移 T_2 位得到 s_2，最终产生的随机数 *rand* 为 s_1 与 s_2 的和，并设 *rand* 为下一次产生随机数的种子；3）最终，对随机数 *rand* 做取余处理从而满足 ViBe 算法的需求。

设置 $T_1 = 4\,164\,903\,690$，$T_2 = 16$，初始化随机数种子 *state* = 0xffff，并将产生的随机数除 10 取余，取前 100 个随机数，表 1 列出了余数 0 ~ 9 出现的次数。

表 1　余数及其出现次数

Table 1　Peminders and their frequency

随机数	0	1	2	3	4	5	6	7	8	9
出现次数	11	10	8	10	8	9	12	10	15	7

理论上，CUDA 架构下的 ViBe 算法时间复杂度为 $O(1)$，但考虑到 GPU 不可能同时执行数万个线程[9]，假设数万个线程由 k 个流处理器乱序执行，那么 GPU 端 ViBe 算法的时间复杂度为 $O(n^2/k)$。

3　实验结果与分析

在 VS2010 平台使用 OpenCV 2.4.10 及 NVIDIA CUDA 7.5 对算法进行并行化处理，实验所用的硬件配置为：Intel Core i5-3230M，NVIDIA GeForce GT 750 M，4.00 GB 内存。

实验中所用的视频序列来自 ChangeDetection 数据集[10]、I2R 数据集[11]，实验所用 ViBe 算法的参数设置为：背景样本大小 $N = 20$，分类阈值 $R = 20$，匹配阈值 #min = 2，时间二次抽样因子 $\varphi = 16$。选取不同分辨率的视频，统计原 ViBe 算法和改进的 ViBe 算法在 CPU 端和 GPU 端耗时及速率对比，结果如表 2 所示，其中，耗时为处理每帧视频所消耗的时间，加速比为 CPU 端与 GPU 端耗时之比，检测速率为每秒钟处理的视频帧数。

表 2　不同分辨率下 CPU 端和 GPU 端检测速率

Table 2　Detection rate of CPU and GPU under different resolutions

分辨率	CPU 端帧速率/(帧·s^{-1})	GPU 端帧速率/(帧·s^{-1})	加速比
320 × 240	328	4182	12.75
640 × 480	97	1392	14.31
768 × 576	64	916	14.23
1280 × 696	24	509	21.02

从表 2 可以看出，GPU 相对 CPU 的加速比将随着分辨率的增加而增加，在 320 × 240 分辨率下，加速比为 12，而在 1280 × 696 分辨率下，加速比为 21。在处理分辨率为 1280 × 696 的监控视频时仍能达到较高的检测速率，足以满足实时处理的需求，可用于实时智能视频监控。

4　结束语

基于 CUDA 的 ViBe 算法在初始化、背景样本更新两个方面对 ViBe 算法进行了改进，从随机数产生、内存及线程分配等方面对 ViBe 算法进行了并行化分析，并用 CUDA 语言实现了 ViBe 算法。实验结果表明，改进的 ViBe 算法可以有效消除边缘噪声、加快消除“鬼影”，检测速度较 CPU 端有大幅提升，检测速度加速比将随着分辨率的增大而增大，具有很高的执行效率，可用于实时智能视频监控等场景。

参 考 文 献

[1] 黄凯奇，陈晓棠，康运锋，等. 智能视频监控技术综述[J]. 计算机学报，2015，38(6)：1093-1118.

[2] BARRON J L, FLEET D J, BEAUCHEMIN S S. Performance of optical flow techniques[C]//IEEE Computer Society Conference on Computer Vision & Pattern Recognition, CVPR, 1992：236-242.

[3] 李刚，曾锐利，林凌，等. 基于帧间颜色梯度的背景建模[J]. 光学精密工程，2007，15(8)：1257-1262.

[4] STAUFFER C, GRIMSON W E L. Adaptive background mixture models for real-time tracking[C]//IEEE Computer Society Conference on Computer Vision and Pattern

（下转第 86 页）

改进时,测试得到的结果会对模拟量采集的后6位数据产生影响,表现为在对同一电压采集时后6位的二进制数据是浮动的。但是,在对A/D单元进行差分形式设计时所得采集数据只有3~4位在浮动,将模拟量采集的精度由10位提高到了12位以上,提高了A/D单元数据采集的精度。

表2 A/D单元单端采集和差分采集转换电压对比

Table 2 Single-ended and differential acquisition conversion voltage of A/D unit

编号	单端采集	差分采集
1	0b1011001100101101	0b1011001100110011
2	0b1011001100100111	0b1011001100110111
3	0b1011001100100111	0b1011001100111001
4	0b1011001100101000	0b1011001100111101
5	0b1011001100100100	0b1011001100110010
6	0b1011001100100001	0b1011001100111110

4.2 A/D单元测试

验证系统内部加入数字滤波器后,对A/D单元进行系统测试。运用美国Xilinx公司所提供的ISE 12.4 ISim对整个系统设计进行了系统测试,所得测试结果如表3所示。

表3 A/D单元加入数字滤波处理数据

Table 3 A/D unit adds digital filter to process data

编号	数字滤波器前	数字滤波器后
1	1011001100111011	1011001100110011
2	1011001100111100	1011001100110010
3	1011001100110111	1011001100110001
4	1011001100111100	1011001100110000
5	1011001100111111	1011001100110011
6	1011001100111010	1011001100110001

从表3可以看出,通过所设计的低通数字滤波器做滤波处理,A/D单元的采集数据有效位提高到14位,且工作稳定,满足无人机飞行控制计算机模拟量数据采集要求,验证了所研究设计的飞行控制计算机A/D单元具有可行性。

5 结论

模拟量采集处理是飞行控制计算机中非常重要的功能,由于现有飞行控制计算机工作环境与电磁环境限制,导致现有转换精度已经接近信号通道的噪声水平。本文对飞行控制计算机的模拟量采集方法进行了研究,设计了差分形式的数据采集系统,介绍了系统的工作原理,描述了模数转换器的电路设计与软件设计,经实验测试,将模拟量采集的精度由10位提高到了14位以上,符合A/D单元数据采集要求,为飞行控制系统AIO单元实现信号的高速、高精度采集提供了一种新思路,本文的研究成果对相关系统测试研究具有参考价值。

参考文献

[1] 徐向荣,陈昌涛,欧阳月华. 飞行控制计算机采集处理系统的设计与实现[J]. 计算机与现代化,2011(2):89-91.

[2] 王琳,商周,王学伟. 数据采集系统的发展与应用[J]. 电测与仪表,2004,41(8):4-8.

[3] 梁葆华,陈欣,吕迅竑. 一种支持双余度CAN总线接口的A/D采集单元设计[J]. 航空计算技术,2008,38(2):107-110.

[4] 马海潮. 超高速数据采集技术发展现状[J]. 测试技术学报,2003,17(4):287-292.

[5] 孙文. 多通道数据采集系统的设计与实现[D]. 长沙:湖南大学,2013.

[6] 许嘉林,卢艳娥,丁子明. ADC信噪比的分析及高速高分辨率ADC电路的实现[J]. 电子技术应用,2004(4):64-67.

[7] 杨延善,苏长莱,梁恺. 航空测控系统实用手册[M]. 北京:航空工业出版社,1997:15-16.

(上接第43页)

Recognition, 1999:246-252.

[5] BARNICH O, VAN DROOGENBROECK M. ViBe: a powerful random technique to estimate the background in video sequences[C]//IEEE International Conference on Acoustics, Speech and Signal Processing, 2009:945-948.

[6] BARNICH O, VAN DROOGENBROECK M. ViBe: a universal background subtraction algorithm for video sequences[J]. IEEE Transactions on Image Processing, 2011, 20(6):1709-1724.

[7] 牛化康,何小海,汪晓飞,等. 一种改进的ViBe目标检测算法[J]. 四川大学学报:工程科学版,2014(s2):104-108.

[8] 徐久强,江萍萍,朱宏博,等. 面向运动目标检测的ViBe算法改进[J]. 东北大学学报:自然科学版,2015,36(9):1227-1231.

[9] 李建江,张磊,李兴钢,等. CUDA架构下的灰度图像匹配并行算法[J]. 电子科技大学学报,2012,41(1):110-113.

[10] GOYETTE N, JODOIN P M, PORIKLI F, et al. Changedetection. net: a new change detection benchmark dataset[C]//Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2012:1-8.

[11] LI L, HUANG W, GU Y H, et al. Statistical modeling of complex backgrounds for foreground object detection[J]. IEEE Transactions on Image Processing, 2004, 13(11):1459-1472.