GPU-based Implementation of an Optimized Nonparametric Background Modeling for Real-time Moving Object Detection

Daniel Berjón, Carlos Cuevas, Francisco Morán, and Narciso García

Abstract — *Answering to the growing demand of computer vision tools for the last generations of consumer electronic devices equipped with smart cameras, several nonparametric moving detection algorithms have been developed. These algorithms, by modeling both background and foreground from spatio-temporal reference data, provide satisfactory results in many complex scenarios. However, to be computationally efficient, they apply some simplifications that decrease the quality of the detections.*

This paper presents a novel real-time implementation of an optimized spatio-temporal nonparametric moving object detection strategy. To improve the quality of previous algorithms, the bandwidths of the kernels required to model the background are dynamically estimated, and the background model is also selectively updated. The proposed implementation features smart cooperation between a computer/device's Central and Graphics Processing Units (CPU/GPU) and extensive usage of the texture mapping and filtering units of the latter, including a novel method for fast evaluation of Gaussian functions. Thanks to these features, high quality detection rates are achieved while respecting the real-time restrictions imposed by computer vision tools running on current consumer electronic devices[1].

Index Terms — Moving object detection, real time, GPU, spatio-temporal nonparametric modeling, smart cameras, high-quality, usability.

I. Introduction

In the latest generation of consumer electronic devices, smart cameras have become very popular among the general public (e.g. mobiles, tablets, or videogame consoles) [1]. As a consequence, many commercial computer vision tools for applications such as augmented reality, classification, and tracking have recently been emerging in the consumer electronic domain at an increasing pace [2]. These tools, as a first and fundamental step, include a moving object detection strategy [3]. Consequently, several approaches proposing moving object segmentation alternatives have been developed in the recent years [4], which are required to be user friendly and to provide high quality results in real time [5].

[1] This work was supported in part by the Ministerio de Economía y Competitividad of the Spanish Government under project TEC2010-20412 (Enhanced 3DTV).

D. Berjón, C. Cuevas, F. Morán, and N. García are with Grupo de Tratamiento de Imágenes (GTI), Universidad Politécnica de Madrid (UPM), 28040 Madrid, Spain (e-mail: {dbd,ccr,fmb,narciso}@gti.ssr.upm.es).

To obtain high-quality detections in complex situations (e.g. dynamic backgrounds, illumination changes, etc.), multimodal moving object detection strategies are commonly used [6], since they are able to model the multiple pixel states in such situations. Among these strategies, nonparametric methods have shown to be those providing the best quality detections [7] because, in contrast to other multimodal algorithms, they do not consider the pixel values as a particular distribution, but obtain instead probabilistic models from sets of recent samples [8].

Most recent nonparametric methods, to improve the quality of the results in sequences containing non-static background regions, and/or recorded with portable cameras, use spatio-temporal reference data [9] and, additionally, model both the background and the foreground variations [10]. However, these strategies involve huge memory and computational costs, since they perform millions of complex operations per image [11]. To achieve the computational efficiency demanded by the latest generations of consumer electronic devices, they carry out some simplifications on the background modeling, which reduce the quality of the detections and worsen the usability of the algorithms.

To begin with, the background is modeled through kernels with fixed bandwidth matrices [9]. Therefore, looking for a compromise between the preservation of the multimodality of the background and the presence of noise in the detections, the user must take care to select proper kernel widths according to the characteristics of each analyzed sequence.

Furthermore, instead of using the selective update mechanisms applied in other moving object detection strategies [12], they use blind algorithms to update the foreground [8]. In this way, they avoid the inclusion of complex selection stages to determine whether a reference sample should or should not be used to model the background. However, the amount of misdetections increases significantly.

Here, we propose an innovative real-time implementation of an optimized background modeling [13], which is suitable for integration in any spatio-temporal nonparametric strategy for moving object detection. To improve the quality and usability of previous approaches, this modeling includes two efficient methods to dynamically estimate the bandwidth matrices of the kernels, and to selectively update the background model. To support these enhancements we have developed an efficient implementation on a consumer-grade Graphics Processing Unit (GPU) featuring some novel techniques to

lower the computational requirements of the algorithm while preserving its quality. The result is a strategy that not only improves the quality of the detections from previous algorithms, but is also suitable for its integration in the last generations of consumer electronic devices demanding real-time computer vision tools.

II. NONPARAMETRIC MOVING OBJECT DETECTION

Let us consider a pixel p^n in the image I^n, at time n, defined as a $(D+2)$-dimensional vector $\mathbf{x}^n=((\mathbf{a}^n)^T,(\mathbf{s}^n)^T)^T \in \mathbb{R}^{D+2}$, where $\mathbf{a}^n \in \mathbb{R}^D$ contains appearance information of the pixel and $\mathbf{s}^n=(h^n,w^n)^T \in \mathbb{R}^2$ contains its coordinates (row and column). The probability of p^n to belong to the sequence foreground, ϕ, can be computed, via Bayes' theorem [14], as

$$\Pr(\phi \mid \mathbf{x}^n)=\frac{\Pr(\phi)p(\mathbf{x}^n \mid \phi)}{\Pr(\beta)p(\mathbf{x}^n \mid \beta)+\Pr(\phi)p(\mathbf{x}^n \mid \phi)}, \quad (1)$$

where $p(\mathbf{x}^n|\beta)$ is the probability density function (pdf) that p^n belongs to the sequence background, β, $p(\mathbf{x}^n|\phi)$ is the pdf that p^n belongs to ϕ, $\Pr(\beta)$ is the background prior probability, and $\Pr(\phi)=1-\Pr(\beta)$ is the foreground prior probability.

Several methods to estimate $p(\mathbf{x}^n|\phi)$ [10] and the prior probabilities [15] have been proposed along the recent years. However, since our work focuses on the quality of the background modeling, we have considered that $p(\mathbf{x}^n|\phi)$ is a constant and that the prior probabilities are $\Pr(\phi)=\Pr(\beta)=½$.

A. Background modeling

Using Gaussian kernels and a set of N_β $(D+2)$-dimensional spatio-temporal reference samples, $\{\mathbf{x}_\beta^i\}_{i=1}^{N_\beta}$, from T_β previous images into a spatial neighborhood [16], the pdf that p^n belongs to β is estimated nonparametrically as

$$p(\mathbf{x}^n \mid \beta)=\frac{1}{N_\beta(2\pi)^{\frac{D+2}{2}}}\sum_{i=1}^{N_\beta}\prod_{j=1}^{D+2}\frac{1}{\left|\boldsymbol{\Sigma}_{\beta,\mathbf{x}}(j,j)\right|^{\frac{1}{2}}}\exp\left(-\frac{\left(\mathbf{x}^n(j)-\mathbf{x}_\beta^i(j)\right)^2}{2\boldsymbol{\Sigma}_{\beta,\mathbf{x}}(j,j)}\right), \quad (2)$$

where $\boldsymbol{\Sigma}_{\beta,\mathbf{x}}$ is a symmetric positive-definite $(D+2)\times(D+2)$ matrix determining the width of the kernels. Looking for a trade-off between computational efficiency and quality [9], this matrix is defined as

$$\boldsymbol{\Sigma}_{\beta,\mathbf{x}}=\mathrm{diag}\left(\sigma_{\beta,1}^2,\sigma_{\beta,2}^2\cdots\sigma_{\beta,\mathrm{D}}^2,\sigma_{\beta,\mathrm{H}}^2,\sigma_{\beta,\mathrm{W}}^2\right), \quad (3)$$

where the first D components determine the bandwidth of the appearance components, and the last two specify the spatial bandwidth of the kernels.

B. Dynamic background bandwidth estimation

As the background reference samples are uniformly distributed in space, we use fixed bandwidths for the spatial components on the whole image extent. However, to achieve the best balance between misdetections and false detections, we estimate dynamically adequate bandwidth values for the appearance components.

Let $\{\Delta\mathbf{a}_\beta^i\}_{i=1}^{N_\beta}$ be the set of D-dimensional appearance differences between all possible consecutive reference samples at the same coordinates. First, for each subset of differences at each spatial position, $\psi_{h,w}=\left\{i \in [1,N_\beta] \text{ such that } \mathbf{s}_\beta^i=(h,w)^T\right\}$, we compute:

$$\boldsymbol{\Sigma}_{\beta,h,w}=\left(\sum_{i\in\psi_{h,w}}\Pr(\beta \mid \mathbf{x}_\beta^i)\right)^{-1/2}\left(\sum_{i\in\psi_{h,w}}\Pr(\beta \mid \mathbf{x}_\beta^i)(\Delta\mathbf{a}_\beta^i)^2\right)^{1/2} \in \mathbb{R}^D, \quad (4)$$

where $\Pr(\beta \mid \mathbf{x}_\beta^i)=1-\Pr(\phi \mid \mathbf{x}_\beta^i)$ is the probability of $\mathbf{x}_\beta^i$ to belong to the sequence background. Weighing the differences by these probabilities the negative influence of outliers is reduced. Moreover, as the probabilities of the reference samples to belong to the background have been previously obtained, they can be used without any additional computational effort.

In a second stage, to take into account the reference data at different spatial coordinates, the bandwidth for each appearance component is finally estimated as

$$\sigma_{\beta,j}=\frac{1}{\sqrt{2}\sum_{h,w}w_\Sigma(h,w)}\sum_{h,w}w_\Sigma(h,w)\boldsymbol{\Sigma}_{\beta,h,w}(j) \text{ such that } j \in [1,D], \quad (5)$$

where $w_\Sigma(h,w)$ is a weight assigned to each subset of differences, according to its spatial distance to p^n. As the influence of the reference samples over each current pixel decreases with their spatial distance to that pixel, these weights are obtained through the evaluation of spatial Gaussians at these distances:

$$w_\Sigma(h,w) \propto \exp\left(-\frac{(h^n-h)^2}{2\sigma_{\beta,H}^2}-\frac{(w^n-w)^2}{2\sigma_{\beta,W}^2}\right) \quad (6)$$

These Gaussians are computed simultaneously with $p(\mathbf{x}^n|\beta)$. Therefore, their use in (5) does not involve any increase of the computational cost.

C. Selective update

To improve the quality of the detections, a very efficient mechanism to selectively update the background model is applied. Thanks to this mechanism, the amount of correctly detected foreground pixels is significantly increased while avoiding most persistent false detections from samples erroneously classified as foreground.

Stemming from (2), the pdf that a current pixel belongs to the image background is estimated as

$$p(\mathbf{x}^n \mid \beta)=K_w\sum_{i=1}^{N_\beta}w_i\prod_{j=1}^{D+2}\exp\left(-\frac{\left(\mathbf{x}^n(j)-\mathbf{x}_\beta^i(j)\right)^2}{2\boldsymbol{\Sigma}_{\beta,\mathbf{x}}(j,j)}\right), \quad (7)$$

where k_w is a normalization factor and w_i is a weight assigned to the i-th reference sample. These weights are estimated in a robust and unsupervised way as

$$w_i=\begin{cases}\Pr(\beta \mid \mathbf{x}_\beta^i) & \text{if } \Delta n_i \le N_C \\ G_T(\Delta n_i)\left(\Pr(\beta \mid \mathbf{x}_\beta^i)-1\right)+1 & \text{if } \Delta n_i > N_C\end{cases}, \quad (8)$$

where Δn_i is the temporal distance between the current sample and the i-th reference one, $N_C \ll T_\beta$ is a predefined constant value, and $G_T(\Delta n_i)$ is a temporal Gaussian defined as

$$G_T(\Delta n_i)=\exp\left(-\frac{(\Delta n_i-N_C)^2}{2\sigma_T^2}\right) \quad (9)$$

To guarantee that this Gaussian accumulates around 99% of its probability in the considered temporal range (i.e. $G_T(T_\beta) \approx 0$) [17], its standard deviation is set as $\sigma_T=(T_\beta-N_C)/3$.

Through this update mechanism, the weights assigned to the reference samples belonging to recently detected foreground objects are very low. Consequently, the negative influence of the moving objects in the background modeling is drastically decreased. Moreover, as the weights assigned to the most recent reference samples are equal to the probabilities of those samples to belong to the background, $\Pr(\beta \mid \mathbf{x}_\beta^i)$, the erroneous classification of foreground objects remaining static for short periods of time is avoided. Additionally, since the temporal Gaussian decreases with the temporal distance from the current sample to the reference ones, all reference samples (regardless of whether they were classified as foreground or background) tend to have more similar weights as this temporal distance increases. Therefore, the amount of false detections resulting from the persistent exclusion of previous samples erroneously classified as foreground is also significantly reduced.

III. REAL-TIME IMPLEMENTATION

The background modeling described in the preceding section involves (using sane parameters) a huge number of operations per pixel, which makes it very slow if run on a regular Central Processing Unit (CPU). Fortunately, it is a very good match with modern programmable GPUs [18]. These devices are becoming commonplace in ordinary computers [19] and even in mobile computing scenarios [20], since they excel in numeric performance and are highly parallel, executing hundreds or thousands of threads concurrently [21]. Logically enough, several approaches for image processing that exploit GPUs have been proposed in the last few years [22], providing real-time implementations of computer vision applications on recent consumer electronic devices [23].

However, to ensure good performance on a GPU, the algorithm or its implementation must comply with some rules that are largely irrelevant on CPUs. GPU designers devote most of the transistors in the die to arithmetic operations in detriment of execution flow control and memory management, just the opposite of a CPU. This makes branching and sparse memory access patterns very expensive. Careful arrangement of memory access instructions and of control structures can, and usually does, make a sizable performance difference on GPUs.

Modern GPUs implement the stream processor paradigm, a form of Single Instruction, Multiple Data (SIMD) parallel processing. Under this paradigm, the same series of operations (kernel function) are independently applied onto each element in a set of data (stream) in an unspecified order and in batches of an (a priori) undetermined number of elements. This computing model is especially suitable for applications exhibiting data parallelism, data locality, and high-compute intensity. The proposed algorithm exhibits all three of these traits:

- Data parallelism: each input pixel is independently classified into foreground or background.

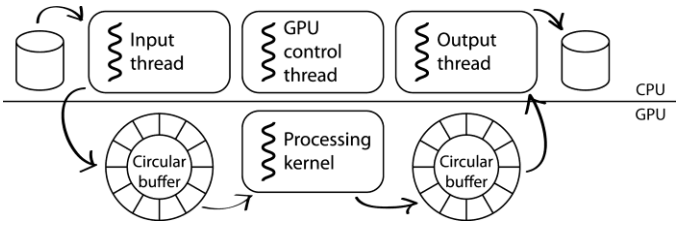


Fig. 1. Multi-threading in the CPU and asynchronous communication among tasks provide low latency and higher throughput.

- Data locality: despite involving many computations, the only end product of the computation is the classification of each pixel; all intermediate results remain within the stream processor, thus reducing memory accesses.
- Compute intensity: each input pixel is employed in thousands of arithmetic operations (note that, although the sheer volume of input data makes it impossible to read from memory each datum only once, caching strategies ensure that it is used many times over).

The current proposal stems from a previous implementation [16] of the most basic spatio-temporal non-parametric background modeling. However, the previous proposal uses some mechanisms (exhaustive lookup table for Gaussians, direct access to global memory for writes) that are not adequate for implementing the enhancements described in Section II to improve the quality of the modeling.

Dynamic bandwidth estimation discourages using exhaustive lookup tables because they should be too large to benefit from caching. It also requires reading and storing differences between successive images in an access pattern that does not lend itself to coalescing memory transactions. In the following subsections we propose novel solutions to these challenges.

A. General architecture

To achieve maximum throughput, the implemented design exploits parallelism not only in the GPU but also in the CPU, offloading input/output operations and their associated computations (decoding and encoding image files) to the host computer. Since input and output operations involve large latencies, separate tasks are created for input, processing control and output; they communicate and pass data along using circular buffer structures that reside in the global memory of the GPU. This asynchronous coupling among the tasks in the CPU, shown in Fig. 1, absorbs the latencies due to disk reading and writing because input data can be read before the processing thread actually needs it, and output data can be written while the processing thread is already busy with the next images.

This architecture also allows us to take full advantage of the capability of the stream processor that we used to perform asynchronous memory transfers from the CPU to the GPU and back in such a way (see Fig. 2) that memory transfers and processing overlap in time. Thus, the GPU processors do not suffer from data starvation and can be fully utilized the whole time, improving the global throughput.

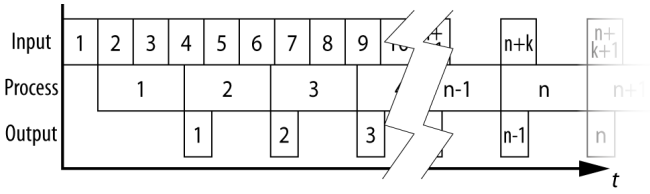


Fig. 2. Asynchronous memory transfers from and to the GPU allow the process task to run continuously, thus maximizing throughput.

B. *Memory access*

The stream processor we used is a consumer-grade GPU that features several different tiers of memory, mainly:

- Global memory: very large (typically over 1 GiB) and accessible from all GPU threads, but located off-chip, yielding relatively low bandwidth and high latency.
- Shared memory: relatively small (e.g. 48 KiB per multiprocessor in the GPU) and only accessible within the same execution block, but located on-chip, yielding very high bandwidth and low latency. It is usually employed as a manually managed cache.
- Constant memory: very small (e.g. 64 KiB total), read-only and accessible from all GPU threads. It is located off-chip but automatically cached, which makes it very fast after the first read.

The implementation we used as a starting point [16] featured heavy use of the shared memory tier to avoid multiple reads from the global memory. However, the limited size of the shared memory has a direct impact in the maximum block sizes and spatial bandwidths that can be used. In fact, as Fig. 3 shows (continuous lines), performance varies depending on block size for a fixed bandwidth because each multiprocessor can schedule fewer thread blocks as their shared memory requirements increase, leading to lower GPU occupancy.

Instead of explicitly caching data in shared memory, we have employed the texture memory both for reading reference data and writing output. Texture memory is not actually a separate tier of memory but rather an access mechanism to the global memory. Instead of uncached access or linear caching that take place (depending on devices) with regular accesses, the texture unit uses space-filling curves such as the Z-order curve [24], pictured in Fig. 4 as an example, to cache the 2-dimensional neighborhood of a memory position. Since our algorithm accesses reference data in the spatial vicinity of each pixel, this memory access is a perfect match and, although it does not guarantee that input pixels will only be read once, it proves indeed faster than [16]. Another beneficial side effect is that processing times are almost independent of the size of processing blocks (see dashed lines in Fig. 3), which improves the usability of the algorithm.

Recent advances in GPU hardware and development tools have allowed us to employ the hardware texturing units and associated 2D caches not only to read data but also to write it, greatly facilitating the storage of difference images, which is crucial to the dynamic appearance bandwidth estimation.

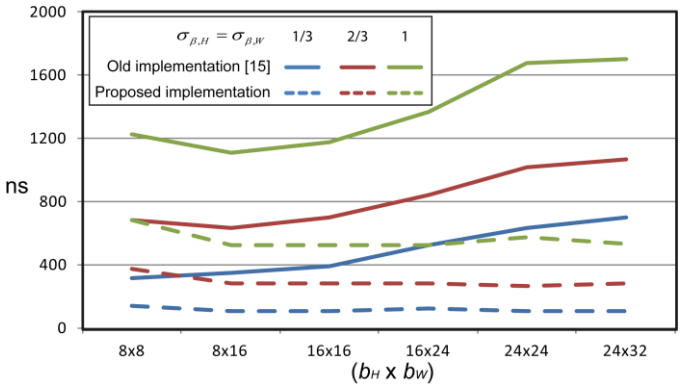


Fig. 3. Mean processing time per pixel, using different spatial bandwidth ($\sigma_{\beta,H}$, $\sigma_{\beta,W}$) and depending on GPU thread block size ($b_H \times b_W$), with fixed appearance bandwidth and no selective update.

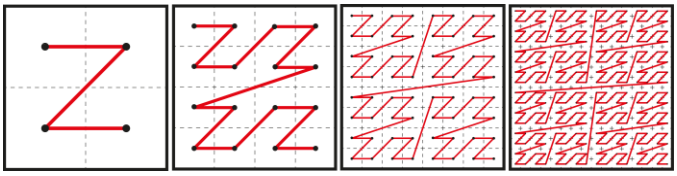


Fig. 4. Four levels of the Z-order curve, an example of space-filling curve that serializes neighboring positions in a 2-dimensional grid.

C. *Gaussian function evaluation*

The proposed background modeling operates over a considerable amount of data and, as defined in Section II, involves D (appearance characteristics) + 2 (spatial coordinates) evaluations of the Gaussian function per datum (~7000 evaluations per input pixel and image with usual parameters [9][16]).

Unfortunately, exponentials are very expensive to evaluate, so this can be a bottleneck for the application. If the variance of the Gaussian is fixed and the range considered is discrete, such as is the case with 8-bit components, all possible evaluations of the Gaussian can be precomputed and stored in constant memory for fast access. However, if the variance is dynamically estimated the number of possible values is too big to efficiently store in a look-up table of exact values; the problem only gets worse if appearance components have a higher resolution, so an alternate method must be found.

To solve this issue, we exploit that any Gaussian can be expressed in terms of the Standard Normal Distribution (SND),

$$f(x,\mu,\sigma)=\frac{1}{\sigma}\mathrm{SND}\left(\frac{x-\mu}{\sigma}\right). \qquad (10)$$

Furthermore, since the Gaussian has a horizontal asymptote in 0, it can be approximated by 0 for $x>c_G$, this being a *cutoff* parameter depending on the specific task at hand. This enables us to take n_G evenly spaced samples in the SND and build a look-up table of user-defined size, independent of variance or data resolution. Evaluation of any Gaussian thus becomes as simple as interpolating values of the look-up table. Fig. 5 shows the absolute value of the relative error between the actual and interpolated value using nearest neighbor or linear

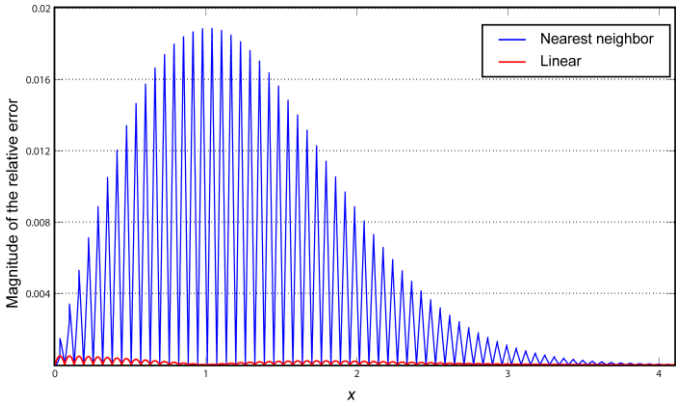


Fig. 5. Absolute value of the relative error between the actual and the interpolated value using nearest neighbor (blue) or linear interpolation (red) for n_G =128 and c_G =8.

interpolation for n_G =128 and c_G=8.

Employing linear interpolation significantly lowers the error for a given number of samples at a modest computational cost. Actually, since we have stored the look-up table as a one-dimensional texture instead of constant memory, the computational cost of the interpolation becomes negligible because linear interpolation is built-in in the circuitry of the texture filtering unit.

Using this technique and the proposed values we obtain 7x speedups compared to actual evaluation of the exponential function while keeping error low enough so that the end results are not affected.

IV. Results

The proposed strategy has been tested in a wide variety of video sequences with different sizes and containing critical aspects for moving object detection such as dynamic backgrounds, illumination changes, non-static cameras, and moving objects remaining static during different periods of time. These sequences, whose main characteristics are summarized in Table I, cover the required indoor and outdoor scenarios and belong to three databases: Lab [25] (Lab_00x), PETS [26] (Pets_00x), and Wallflower [27] (Wall_00x). In addition, to provide an overall quality result for some analysis stages, we have also considered the union of all test sequences (Test-Set).

Since the proposed strategy is focused on the quality of the background modeling, to help analyze the obtained improvements, we have considered in all experiments that there is no previous knowledge (i.e. Pr(β)=Pr(ϕ)=½) and that the foreground pdf is constant.

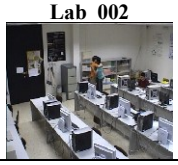
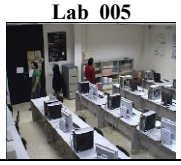
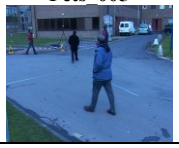
The background modeling has been carried out by using T_β=150 reference images. We have set the spatial bandwidths of the Gaussians kernels to consider neighborhoods of 8-connected pixels, since these widths are enough to avoid most false detections due to small background displacements, and larger spatial widths would significantly increase the computational cost of the modeling [16]. To avoid the negative influence of moving objects remaining static for short periods of time, we have chosen to use N_C=25 images to selectively update the background model.

Therefore, free of the constraint to use the typical low-precision RGB color components, we have been able to employ the appearance vector proposed in [25], which is gradient of the brightness, $|\nabla s|$. Thanks to this, the negative influence of shadows and reflected light in the detections is composed by the chromaticity, (Rn,Gn), and the module of the decreased considerably.

The proposed strategy has been compared, in terms of quality and computational efficiency, to the ones in [9] and [28], because:

- The background modeling in [9] (whose GPU-based real-time implementation is described in [16]) can be considered the fastest and simplest nonparametric spatio-temporal approach, since it uses fixed diagonal bandwidth matrices, a

TABLE I
Description of the Test Sequences

	Lab_001	Lab_002	Lab_003	Lab_004	Lab_005	Pets_001
						
No. images	325	380	550	500	250	1452
Duration (s)	13	15	22	20	10	58
Size (H×W)	288×352	288×352	192×256	196×256	288×352	288×384
No. moving objects	1	2	1	1	2	8
No. ground truth images	15	16	22	21	11	32
	Pets_002	**Pets_003**	**Pets_004**	**Pets_005**	**Wall_001**	**Wall_002**
						
No. images	500	435	795	795	293	287
Duration (s)	20	17	31	31	11	11
Size (H×W)	288×384	288×384	288×352	288×352	128×160	128×160
No. moving objects	0	4	26	27	1	1
No. ground truth images	500	18	31	31	293	287

blind update mechanism, and RGB appearance information. Consequently, because of simplicity and computational requirements, several moving object detection strategies [10] [25] [29] [30] [31] take it as starting point.

- The strategy in [28] is a very efficient CPU-based implementation of the classical MoG method proposed in [32], a seminal work that has been taken as main reference by hundreds of moving object detection strategies [33].

A. Quality analysis

To measure objectively the quality of the detections, the conventional recall (*rec*), precision (*pre*), and *F* evaluation parameters [34] have been used:

$$rec = 100\frac{cd}{cd+nd}\%, \quad pre = 100\frac{cd}{cd+fd}\%, \quad F = 2\frac{rec \times pre}{rec+pre}\%, \quad (11)$$

where *cd* is the amount of correct detections, *nd* is the number of misdetections, and *fd* is the number of false detections.

Table II summarizes the recall, precision and *F* percentages obtained with all the strategies evaluated, while Fig. 7 illustrates representative detections obtained in some sequences. It must be noted that the results corresponding to the background modeling in [9] have been obtained by using the best global appearance bandwidth value we have found and the same spatial width we are using in our strategy.

These results show that the MoG-based method is able to obtain high recall percentages in most sequences. However, it does not model correctly the background changes in many sequences, especially in those with very dynamic background (first image in Fig. 7.c) or fast illumination changes (fourth image in Fig. 7.c). Therefore, it results in a huge amount of false detections that decrease the global quality (very low F percentages). Additionally, as it uses RGB color components in the modeling, it yields large amounts of false detections due to shadows and reflected light.

Besides, the obtained results prove that the fixed bandwidth values used by the background modeling in [9] not only decrease the usability, but also make it difficult to maintain both high recall and precision percentages in all sequences, and even inside a given sequence. Additionally, as the background model is updated with a blind mechanism, in sequences with objects moving along the optical axis (second image in Fig. 7.d) or remaining static for very short periods of time (third image in Fig. 7.d), many misdetections occur. Moreover, as for the MoG-based method and for the same reasons, it does not avoid false detections due to shadows and reflected light.

TABLE II
QUALITY ANALYSIS: SUMMARY OF RECALL, PRECISION, AND *F* PERCENTAGES OBTAINED WITH THE STRATEGIES EVALUATED

	MoG-based strategy in [28]			Background modeling in [9]			Proposed modeling		
	rec	*pre*	*F*	*rec*	*pre*	*F*	*rec*	*pre*	*F*
Lab_001	73	84	78	58	72	64	85	86	86
Lab_002	76	84	80	47	79	59	63	86	73
Lab_003	78	67	72	23	74	35	34	87	49
Lab_004	84	69	76	62	88	73	64	93	76
Lab_005	69	84	76	59	72	65	77	83	80
Pets_001	81	73	77	35	72	47	51	84	63
Pets_002	100	0	0	100	0	0	100	0	0
Pets_003	74	70	72	57	72	63	73	84	78
Pets_004	75	56	64	37	87	52	27	92	42
Pets_005	82	79	80	52	87	66	33	94	48
Wall_001	92	48	63	19	45	27	68	92	78
Wall_002	94	7	13	77	16	26	85	89	87
Test-Set	85	18	30	38	39	38	54	86	66

Results in Table II and Fig. 7 show as well that the strategy we propose obtains the best global quality detections. By dynamically estimating the kernel bandwidth values, it significantly reduces the number of false detections in all sequences; and by selectively updating the background model, it increases the amount of correct detections. Additionally, by using the chromaticity and the gradient of the brightness as the appearance characteristics of the pixels, it avoids most false detections due to shadows and reflected light.

Although the proposed modeling provides the best quality in most sequences, it can be observed that in Pets_004 and Pets_005 the strategies in [9] and [16] obtain better quality. In these sequences the moving objects remain static or move along the optical axis for several seconds, which leads to several misdetections. As described in Subsection II.B, the proposed bandwidth estimation depends on previous detections. Therefore, the quality of the estimation decreases as the number of misdetections increases.

B. Computational analysis

The proposed strategy has been implemented and measured on a consumer-grade GPU with 16 stream multiprocessors and 1.5 GiB of RAM (estimated global computing power of 1581.1 GFLOPs), coupled with a 4-core CPU clocked at 3.4 GHz with 16 GiB RAM.

The achieved computational efficiency has been compared to those obtained by the strategy in [9] on the same GPU, and by the strategy in [28] on a 2.66 GHz CPU with 2 GiB of RAM.

In the first place, using the GPU-based scheme proposed in this paper we have analyzed both the computational efficiency of the most basic spatio-temporal nonparametric background modeling [9], and the computational cost increase after adding to that model the improvements proposed in Section II and replacing the RGB color components with the appearance vector (Rn,Gn,$|Vs|$). Fig. 6 shows a summary of the mean computational costs per pixel (once the background model has been fully initialized) resulting from this analysis.

Variations in mean processing time (typically under 10%) are explained by the peculiar distribution of differences between consecutive pixels in each sequence. Although the operations performed for each pixel in every sequence are exactly the same, the data are not. This induces a unique hit pattern on the texture and associated caches that store the values of the Gaussian functions. Each of the proposed enhancements alters the appearance bandwidth of each pixel in each instant, which explains that mean processing times using different strategies are not proportional across different sequences.

Finally, we have compared the computational efficiency of evaluated methods. The obtained processing times, once the background model has been fully initialized, appear in

Fig. 7. Representative final detections obtained with different moving object detection strategies. (a) Original images. (b) Ground truth images. (c) Typical MoG-based moving object detection strategy [28]. (d) Basic spatio-temporal nonparametric background modeling [9]. (e) Proposed strategy.

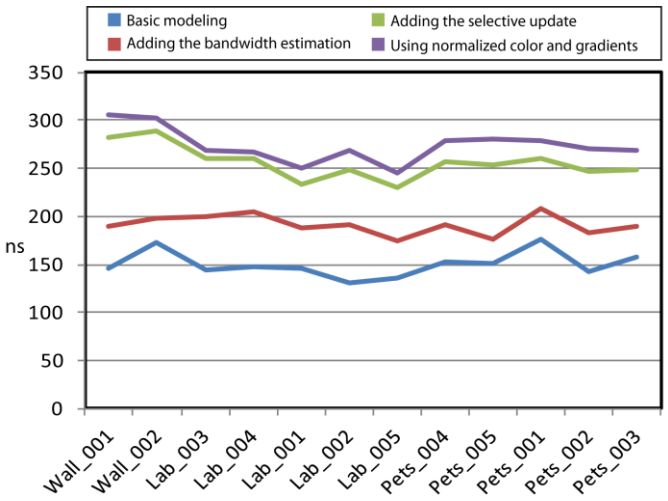


Fig. 6. Mean computational time per pixel in the most basic spatio-temporal nonparametric modeling (blue), and after adding the proposed improvements (red, green, and purple).

Table III (sorted by increasing image-size of the sequences). The second column shows the results corresponding to the MoG-based method proposed in [28], which was designed to provide real-time results on a CPU. The third column presents the results obtained with the spatio-temporal nonparametric background modeling in [9], which was also originally designed to be implemented on a CPU. The fourth column contains the times with the GPU-based implementation [16] of the nonparametric strategy in [9]. The fifth and sixth columns list the processing times and corresponding frame rates achieved with the proposed strategy. It can be observed that we not only maintain but also in fact improve the computational efficiency achieved with the GPU-based implementation in [16].

Accordingly, we are able to provide real-time detections, even for the sequences with the highest image sizes. If we compare our results with those of the CPU-based implementations, we observe that we reduce more than 200 times the processing times achieved in [9], and that we get times similar to those reached in [28]. However, it should be noted that the computational cost achieved with [28] is not proportional to the image sizes and highly depends on the amount of foreground pixels in each moment. Therefore, the processing time achieved by the strategy in [28] is not constant and, consequently, in presence of large amounts of

TABLE III
MEAN PROCESSING TIMES (MS) OF THE STRATEGIES EVALUATED AND SPEED (FPS) ACHIEVED WITH THE PROPOSED MODELING

Seq.	MoG (CPU)	NPM (CPU)	NPM (GPU)	Proposed NPM (GPU)	Speed (fps)
Wall_001	13	1697	9	6	167
Wall_002	14	1707	10	6	167
Lab_003	13	3958	19	13	75
Lab_004	14	3972	18	13	75
Lab_001	25	8551	38	25	38
Lab_002	27	8415	38	27	37
Lab_005	25	9397	38	25	40
Pets_004	30	8549	35	28	35
Pets_005	32	8517	35	31	35
Pets_001	30	8849	37	30	33
Pets_002	29	9162	39	29	34
Pets_003	28	9181	37	30	34

moving objects, it can result in several detections missed, which would not happen with the strategy we propose here.

V. CONCLUSION

An innovative real-time GPU-based implementation of a high-quality background modeling has been presented, which is suitable for integration in any nonparametric spatio-temporal moving object detection strategy, such as the ones demanded by the computer vision applications running on current consumer electronic devices.

By selectively updating the background model through a robust and unsupervised estimation of weights assigned to each reference sample, the number of misdetections is significantly reduced. Additionally, adequate bandwidth matrices are dynamically estimated from spatially weighted sample variances resulting from distributions of differences of reference data. In this way, an additional improvement of the quality is achieved and, moreover, the usability of previous approaches is also improved.

The proposed implementation features multitasking and asynchronous cooperation between CPU and GPU, thereby preventing data starvation and increasing device utilization. Reading and writing data through the hardware texture caches have proven very useful for this algorithm, providing higher parallelism degrees than previous explicit cache approaches and high performance regardless of the thread block size. Finally, since the evaluation of Gaussian functions consumes a very significant portion of the computation time, we have also proposed a novel technique to accomplish it, which exploits hardware features of the GPU such as its texture mapping and filtering units.

The obtained results have shown that the proposed strategy not only enhances the quality and usability of previous algorithms but also provides real-time results in a large variety of sequences with different image-sizes.

REFERENCES

[1] K. Sangani, "2010 gadget census [Consumer Tech Census]," *Engineering & Technology*, vol. 5, no. 14, pp. 28-29, 2010.

[2] W. La, J. Han, and P. H. N. de With, "Automatic video-based human motion analyzer for consumer surveillance system," *IEEE Trans. Consumer Electronics*, vol. 55, no. 2, pp. 591-598, 2009.

[3] P. Chiranjeevi and S. Sengupta, "Robust detection of moving objects in video sequences through rough set theory framework," *Image and Vision Computing*, vol. 30, no. 11, pp. 829-842, 2012.

[4] H. Hassanpour, M. Sedighi, and A. Manashty, "Video frame's background modeling: Reviewing the techniques," *Journal of Signal and Information Processing*, vol. 2, no. 2, pp. 72-78, 2011.

[5] Q. Wnag, W. Zeng, and A. G. Lobzhanidze, "Mobile media in action: Remote target localization and tracking," *IEEE Multimedia*, vol. 19, no. 3, pp. 74-80, 2012.

[6] H. Lin, J. Chuang, and T. Liu, "Regularized background adaptation: a novel learning rate control scheme for Gaussian mixture modeling," *IEEE Trans. Image Processing*, vo. 20, no. 3, pp. 822-836, 2011.

[7] A. Tavakkoli, M. Nicolescu, G. Bebis, and M. Nicolescu, "Non-parametric statistical background modeling for efficient foreground region detection," *Machine Vision and Applications*, vol. 20, no. 6, pp. 395-409, 2009.

[8] A. Mittal and N. Paragios, "Motion-based background subtraction using adaptive kernel density estimation," *IEEE Int. Conf. Computer Vision and Pattern Recognition*, vol. 2, pp. 302-309, 2004.

[9] Y. Sheikh and M. Shah, "Bayesian modeling of dynamic scenes for object detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, no.11, pp. 1778-1792, 2005.

[10] X. Zhang and J. Yang, "Foreground segmentation based on selective foreground model," *Electronics Letters*, vol. 44, no. 14, pp. 851-852, 2008.

[11] S. Elhabian, K. El-Sayed, and S. Ahmed, "Moving object detection in spatial domain using background removal techniques-state-of-art," *Recent Patents on Computer Science*, vol. 1, no.1, pp. 32-54, 2008.

[12] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, "Detection moving objects, ghosts, and shadows in video streams," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vo. 25, no. 10, pp. 1337-1342, 2003.

[13] C. Cuevas and N. García, "Improved background modeling for real-time spatio-temporal non-parametric moving object detection," *Image and Vision Computing* (under review).

[14] C. Cuevas, R. Mohedano, and N. García, "Adaptable Bayesian classifier for spatio-temporal non-parametric moving object detection strategies," *Optics Letters*, vol. 37, no. 15, pp. 3159-3161, 2012.

[15] N. Martel-Brison and A. Zaccarin, "Unsupervised approach for building non-parametric background and foreground models of scenes with significant foreground activity," *ACM Workshop on vision Networks for Behavior Analysis*, pp. 93-100, 2008.

[16] C. Cuevas, D. Berjón, F. Morán, and N. García, "Moving object detection for real-time augmented reality applications in a GPGPU," *IEEE Trans. Consumer Electronics,* vol. 58, no. 1, pp. 117-125, 2012.

[17] Z. Guo, G. Jiang, H. Chen, and K. Yoshihira, "Tracking probabilistic correlation of monitoring data for fault detection," *IEEE Int. Conf. Complex Systems and Networks*, pp. 259-268, 2006.

[18] T. D. Han and T.S. Abdelrahman, "hicuda: High-level GPGPU programming," *IEEE Trans. Parallel and Distributed Systems*, vol. 22, no. 1, pp. 78-90, 2011.

[19] B. Neelima and P. S. Raghavendra, "Recent trends in software and hardware for GPGPU computing: a comprehensive survey," *IEEE Int. Conf. Industrial and Information Systems*, pp. 319-324, 2010.

[20] S. W. Keckler, W. J. Dally, B. Khailany, M. Garland, and D. Glasco, "GPUs and the future of parallel computing," *IEEE Micro*, vol. 31, no. 5, pp. 7-17, 2011.

[21] S. F. Tsai, C. C. Cheng, C. T. Li, and L. G. Chen, "A real-time 1080p 2D-to-3D video conversion system," *IEEE Trans. Consumer Electronics*, vol. 57, no. 2, pp. 915-922, 2011.

[22] G. Bravo, S. Zinger, and P. H. N. de With, "GPU-accelerated real-time free-viewpoint DIBR for 3DTV," *IEEE Trans. Consumer Electronics*, vol. 58, no. 2, pp. 633-640, 2012.

[23] J. Park, J. Y. Choi, I. Ryu, and J. I. Park, "Universal view synthesis unit for glassless 3DTV," *IEEE Trans. Consumer Electronics*, vol. 58, no. 2, pp. 706-711, 2012.

[24] M. Doggett, "Texture caches," *IEEE Micro*, vol. 32, no. 3, pp. 136-141, 2012.

[25] C. Cuevas and N. García, "Efficient moving object detection for lightweight applications on smart cameras," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 23, no. 1, pp. 1-14, 2013.

[26] Computational Vision Group, "Pets: Performance evaluation of tracking and surveillance," *University of Reading*, England.

[27] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, "Wallflower: Principles and practice of background maintenance," *IEEE Int. Conf. Computer Vision*, vol. 1, pp. 255-261, 1999.
[28] C. Cuevas, N. García, and L. Salgado, "A new strategy base don adaptive mixture of Gaussians for real-time moving objects segmentation," *SPIE Real-Time Image Processing*, vol. 6811, pp. 681111-1-12, 2008.
[29] T. Ko, S. Soatto, and D. Estrin, "Background subtraction on distributions," *European Conf. Computing Vision*, pp. 276-289, 2008.
[30] Y. Sheikh, O. Javed, and T. Kanade, "Background subtraction for freely moving cameras," *IEEE Int. Conf. Computer Vision*, pp. 1219-1225, 2009.
[31] X. Zhang, J. Yang, and Z. Liu, "Foreground segmentation based on tracking," *Optical Engineering*, vol. 48, no. 10, pp. 7203, 2009.
[32] C. Stauffer and W. E. L. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 747-757, 2000.
[33] T. Bouwmans, F. El Baf, and B. Vachon, "Background modeling using misture of Gaussians for foreground detection-a survey," *Recent Patents on Computer Science*, vol. 1, no. 3, 219-237, 2008.
[34] F. C. Chen and S. J. Ruan, "Accurate motion detection using a self-adaptive background matching framework," *IEEE Trans. Intelligent Transportation Systems*, vol. 13, no. 2, pp. 671-679, 2012.

BIOGRAPHIES

Daniel Berjón received the Ingeniero de Telecomunicación degree (integrated BSc-MSc accredited by ABET) from the Universidad Politécnica de Madrid (UPM), Spain, in 2005, and is currently a student of UPM's PhD in Communications program.
Since 2008 he is a member of UPM's Grupo de Tratamiento de Imágenes (Image Processing Group). His research interests include computer graphics, parallel processing and real-time systems.

Carlos Cuevas received the Ingeniero de Telecomunicación degree (integrated BSc-MSc accredited by ABET) in 2006 and the Doctor Ingeniero de Telecomunicación degree (PhD in Communications) in 2011, both from the Universidad Politécnica de Madrid (UPM), Spain.
Since 2006 he is a member of UPM's Grupo de Tratamiento de Imágenes (Image Processing Group). His research interests include signal and image processing, computer vision, pattern recognition and automatic target recognition.

Francisco Morán received the Ingeniero de Telecomunicación degree (six years engineering curriculum) in 1992 and the Doctor Ingeniero de Telecomunicación degree (PhD in Communications) in 2001, both from the Universidad Politécnica de Madrid (UPM), Spain.
Since 1992 he is a researcher at UPM's Grupo de Tratamiento de Imágenes (Image Processing Group), and since 1997 a member of UPM's faculty. His research interests include hierarchical modeling and coding, and adaptive transmission and visualization of 3D objects. He has been actively involved in research projects funded by the European Union, and participates since 1996 in the standardization activities from ISO's Moving Picture Experts Group (MPEG), where he is the Head of the Spanish Delegation since 2006, and has served as (co-)editor of eight standards, amendments and corrigenda related to MPEG-4 (formally, ISO/IEC 14496).

Narciso García received the Ingeniero de Telecomunicación degree (five years engineering curriculum) in 1976 (Spanish National Graduation Award) and the Doctor Ingeniero de Telecomunicación degree (PhD in Communications) in 1983 (Doctoral Graduation Award), both from the Universidad Politécnica de Madrid (UPM), Spain.
Since 1977 he is a member of UPM's faculty, currently a Professor of Signal Theory and Communications. He leads UPM's Grupo de Tratamiento de Imágenes (Image Processing Group). He has been actively involved in Spanish and European research projects, serving also as evaluator, reviewer, auditor, and observer of several research and development programmes of the European Union. He was a co-writer of the EBU proposal, base of the ITU standard for digital transmission of TV at 34-45 Mb/s (ITU-T J.81). His professional and research interests are in the areas of digital image and video compression and of computer vision.