Computer Vision and Image Understanding 122 (2014) 65–73

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

The *3dSOBS+* algorithm for moving object detection

Lucia Maddalena [a,*], Alfredo Petrosino [b,1]

[a] National Research Council, Institute for High-Performance Computing and Networking, Via P. Castellino 111, 80131 Naples, Italy
[b] University of Naples Parthenope, Department of Science and Technology, Centro Direzionale, Isola C/4, 80143 Naples, Italy

ARTICLE INFO

Article history:
Received 16 April 2013
Accepted 22 November 2013

Keywords:
Background subtraction
Motion detection
Neural network
Self organization

ABSTRACT

We propose the *3dSOBS+* algorithm, a newly designed approach for moving object detection based on a neural background model automatically generated by a self-organizing method. The algorithm is able to accurately handle scenes containing moving backgrounds, gradual illumination variations, and shadows cast by moving objects, and is robust against false detections for different types of videos taken with stationary cameras. Experimental results and comparisons conducted on the Background Models Challenge benchmark dataset demonstrate the improvements achieved by the proposed algorithm, that compares well with the state-of-the-art methods.

1. Introduction

Several methods have been proposed for moving object detection based on background subtraction [1–4]. Nonetheless, the best approach has not yet been devised, as also witnessed by recent activities aimed at comparing the state-of-the-art methods on publicly available video sequence datasets [5,6].

Moving object detection, as a specific case of object segmentation, can be viewed as a clustering problem in a feature space derived from the color and motion information, and therefore is well suited for unsupervised modeling approaches. Unsupervised learning is in general preferred over supervised learning because the latter requires a set of training samples, which may not be available, especially when the image features are unknown or when a certain degree of automation is desired [7].

Many approaches have been explored, including statistical background modeling (e.g., single Gaussian [8] or Mixture of Gaussians [9]), clustering-based background modeling (e.g., Codebook [10]), neural-based background modeling (e.g., [11]), eventually corroborated by the introduction of fuzzy concepts to handle imprecision and uncertainties [12,13]. Neural network-based solutions have received considerable attention due to the fact that these methods are usually more effective and efficient than traditional ones, relying on the well-known advantages of neural networks, such as adaptivity, parallelism, and learning [14]. Among the most recent results, the 3D Self-Organizing Background Subtraction algorithm [15] implements an approach for moving object detection based on a neural background model automatically generated by a self-organizing method, that accurately handles most of the well known background maintenance issues.

The main contribution of our paper relies on an enhanced version of the algorithm concerning the introduction of: (a) initial background estimation; (b) shadow detection and removal; and (c) spatial coherence. These enhancements lead to the *3dSOBS+* algorithm that can accurately handle scenes containing moving backgrounds, gradual illumination variations, and shadows cast by moving objects, and provides further robustness against false detections for different types of videos taken with stationary cameras. Experimental results and comparisons conducted on the Background Models Challenge (BMC) benchmark dataset [6] demonstrate the improvements achieved by the algorithm and that it compares well with the state-of-the-art methods.

The paper is organized as follows. In Section 2 we describe in a unified framework the neural background model adopted from [15] for moving object detection and its enhanced version here proposed, while in Section 3 we provide a thorough analysis of experimental results on the BMC dataset and comparisons with several state-of-the-art methods. Conclusions are drawn in Section 4.

2. The *3dSOBS+* algorithm

The proposed algorithm relies on our main idea to adopt a biologically inspired problem-solving method based on visual attention mechanisms. The aim is to obtain objects that keep the user attention in accordance with a set of predefined features, including gray level, motion, and shape features. Our approach defines a method for the generation of an active attention focus to monitor

* Corresponding author. Fax: +39 081 6139531.
E-mail addresses: lucia.maddalena@cnr.it (L. Maddalena), alfredo.petrosino@uniparthenope.it (A. Petrosino).
1 Fax: +39 081 5476514.

1077-3142/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.cviu.2013.11.006

dynamic scenes for surveillance purposes. The idea is to build the background model by learning in a self-organizing manner many background variations. Based on the learned background model through a map of motion and stationary patterns, our algorithm detects moving objects and selectively updates the background model. The obtained self-organizing neural network can be organized as a 2D grid of neurons [11] or a 3D grid of neurons [15], in both cases producing a representation of training samples with lower dimensionality, at the same time preserving topological neighborhood relations of the input patterns.

In the following we will describe the algorithm and how it is able to tackle the most frequent and difficult background maintenance problems [16], including bootstrapping, illumination changes, waving trees, cast shadows, and robustness against false detections.

2.1. Neural background model representation

Given an image sequence $\{I_t\}$, for each pixel $\mathbf{x}$ in the image domain, a neural map is built, consisting of n weight vectors $m_t^i(\mathbf{x}), i = 1, \ldots, n$, which will be called a *model* for pixel $\mathbf{x}$. If each sequence frame has N rows and P columns, the complete set of models $M_t(\mathbf{x}) = (m_t^1(\mathbf{x}), \ldots, m_t^n(\mathbf{x}))$ for all pixels $\mathbf{x}$ of the t-th sequence frame I_t is organized as a 3D neural map $\mathcal{B}_t$ with N rows, P columns, and n layers. Therefore, the neural background model $\mathcal{B}_t$ consists of n images $L_t^i, i = 1, \ldots, n$, of the same size of image I_t, which we call *layers*, and each layer L_t^i contains, for each pixel $\mathbf{x}$, the i-th weight vector $m_t^i(\mathbf{x})$. A pictorial representation of this 3D neural model can be found in [15].

2.2. Neural background model initialization

In [15], the neural model is initialized by setting all the model layers equal to the first frame of the sequence. However, in order to better handle cases where also in the initial frames the background is occluded by foreground objects (the so-called *bootstrapping* problem [16]), in this paper we propose to adopt one of the methods for background estimation, such as those proposed in [17] and references therein, to achieve an initial estimate E_0 of the scene background. As an example, in our experiments we adopt the temporal median method [18], that consists in estimating the initial background E_0 as the temporal median over a subset of F initial sequence frames $I_0, \ldots, I_{F-1}$. Then, the neural background model $\mathcal{B}_0$ is initialized by setting all weight vectors related to a pixel $\mathbf{x}$ equal to the pixel brightness value of the estimated background, that is,

$$m_0^i(\mathbf{x}) = E_0(\mathbf{x}), \quad i = 1, \ldots, n. \tag{1}$$

Therefore, the resulting initial neural map $\mathcal{B}_0$ consists of n layers, each of which is a copy of the estimated background E_0.

2.3. Background subtraction and neural background model update

After initialization, at each time step t, background subtraction is achieved by comparing each pixel $\mathbf{x}$ of the tth sequence frame I_t with the pixel current model $M_{t-1}(\mathbf{x})$, in order to determine if there exists a best matching weight vector $m_{t-1}^b(\mathbf{x})$ that is close enough to it. If no acceptable matching weight vector exists, $\mathbf{x}$ is detected as foreground; otherwise, $\mathbf{x}$ is detected as a background pixel. In case of background pixels, further learning of the neural map enables the adaptation of the background model to slight scene modifications, such as gradual illumination changes. This learning is achieved by updating the neural weights according to a visual attention mechanism of reinforcement, where the best matching weight vectors, together with their neighborhood, are reinforced into the neural map.

The above described steps will be detailed in the following subsections, also extending the background model update in order to enhance robustness against false detections and to better handle cast shadows.

2.3.1. Finding the best match

At time t, the value $I_t(\mathbf{x})$ of each incoming pixel $\mathbf{x}$ of the t-th sequence frame I_t is compared to the pixel current model $M_{t-1}(\mathbf{x}) = (m_{t-1}^1(\mathbf{x}), \ldots, m_{t-1}^n(\mathbf{x}))$, to determine the weight vector $m_{t-1}^b(\mathbf{x})$ that best matches it:

$$d(m_{t-1}^b(\mathbf{x}), I_t(\mathbf{x})) = \min_{i=1,\ldots,n} d(m_{t-1}^i(\mathbf{x}), I_t(\mathbf{x})), \tag{2}$$

where the metric $d(\cdot,\cdot)$ is suitably chosen according to the specific color space being considered. For the experiments reported in Section 3, the Euclidean distance of vectors in the HSV color hexcone, as suggested in [19], has been adopted, that gives the distance between a weight vector $m_{t-1}^i(\mathbf{x})$ and a pixel value $I_t(\mathbf{x})$ as:

$$\begin{aligned} d(m_{t-1}^i(\mathbf{x}), I_t(\mathbf{y})) = \|&(m^V \cdot m^S \cdot \cos(m^H), m^V \cdot m^S \cdot \sin(m^H), m^V) \\ &- (I^V \cdot I^S \cdot \cos(I^H), I^V \cdot I^S \cdot \sin(I^H), I^V)\|_2^2, \end{aligned} \tag{3}$$

where (m^H, m^S, m^V) and (I^H, I^S, I^V) indicate the hue, saturation, and value components of $m_{t-1}^i(\mathbf{x})$ and $I_t(\mathbf{x})$, respectively.

2.3.2. Updating the model

In order to adapt the neural background model to scene modifications, the model $M_{t-1}(\mathbf{x})$ for pixel $\mathbf{x}$ at time t should be updated. The best matching weight vector $m_{t-1}^b(\mathbf{x})$ (computed according to Eq. (2)) and its neighboring weight vectors of the b-th layer of model $\mathcal{B}_t$ are updated according to weighted running average:

$$m_t^b(\mathbf{y}) = (1 - \alpha(\mathbf{x},\mathbf{y}))m_{t-1}^b(\mathbf{y}) + \alpha(\mathbf{x},\mathbf{y})I_t(\mathbf{y}), \quad \forall \mathbf{y} \in N_{\mathbf{x}}. \tag{4}$$

Here $N_{\mathbf{x}} = \{\mathbf{y} : |\mathbf{x} - \mathbf{y}| \leqslant w_{2D}\}$ is a 2D square spatial neighborhood of $\mathbf{x}$ with half-width w_{2D} including $\mathbf{x}$. Moreover,

$$\alpha(\mathbf{x},\mathbf{y}) = \gamma \cdot G_{2D}(\mathbf{y} - \mathbf{x}) \cdot (1 - D_t(\mathbf{x})) \cdot (1 - S_t(\mathbf{x})), \tag{5}$$

where γ represents the learning rate, $G_{2D}(\cdot) = \mathcal{N}(\cdot; \mathbf{0}, \sigma_{2D}^2 I)$ is a 2D Gaussian low-pass filter [20] with zero mean and $\sigma_{2D}^2 I$ variance, $D_t(\mathbf{x})$ indicates the background subtraction mask value for pixel $\mathbf{x}$, that will be described in Section 2.3.3, and $S_t(\mathbf{x})$ is the shadow mask value indicating if pixel $\mathbf{x}$ belongs to the shadow cast by an object in the scene, that will be described in Section 2.3.4. The $\alpha(\mathbf{x},\mathbf{y})$ values in Eq. (5) are weights that allow the neural model to smoothly take into account the spatial relationship between current pixel $\mathbf{x}$ and its neighboring pixels $\mathbf{y} \in N_{\mathbf{x}}$, thus preserving topological properties of the input (close inputs correspond to close outputs). The choice of the learning factor γ depends on the scene variability: large γ values enable the network to faster learn changes corresponding to the background, but also leading to false negatives, that is, inclusion into the background model of pixels belonging to foreground moving objects. On the contrary, lower learning rates make the network slower to adapt to rapid background changes, making the model more tolerant to errors due to false negatives through self-organization. Indeed, weight vectors of false negative pixels are readily smoothed out by the learning process itself.

The 2D update of Eq. (4) involves only model weight vectors lying in the same layer b as the best matching weight vector $m_{t-1}^b(\mathbf{x})$. In order to further enable the reinforcement of $m_{t-1}^b(\mathbf{x})$ in the model for pixel $\mathbf{x}$, also weight vectors of $\mathbf{x}$ belonging to layers close to layer b are updated. This further update is achieved by weighted running average:

$$m_t^i(\mathbf{x}) = (1 - \delta(\mathbf{x}))m_{t-1}^i(\mathbf{x}) + \delta(\mathbf{x})I_t(\mathbf{x}), \tag{6}$$