Computer Vision and Image Understanding 115 (2011) 105–116

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

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

Unsupervised learning of background modeling parameters in multicamera systems

Konstantinos Tzevanidis[a,b], Antonis Argyros[a,b,*]

[a] Computer Science Department, University of Crete, Knossou Ave., GR 71409 Heraklion, Crete, Greece
[b] Institute of Computer Science, FORTH, N. Plastira 100, Vassilika Vouton, GR 70013, Heraklion, Crete, Greece

ARTICLE INFO

Article history:
Received 16 March 2010
Accepted 19 September 2010
Available online 25 October 2010

Keywords:
Background modeling
Foreground detection
Multicamera consensus
Particle Swarm Optimization
Camera networks

ABSTRACT

Background modeling algorithms are commonly used in camera setups for foreground object detection. Typically, these algorithms need adjustment of their parameters towards achieving optimal performance in different scenarios and/or lighting conditions. This is a tedious process requiring considerable effort by expert users. In this work we propose a novel, fully automatic method for the tuning of foreground detection parameters in calibrated multicamera systems. The proposed method requires neither user intervention nor ground truth data. Given a set of such parameters, we define a fitness function based on the consensus built from the multicamera setup regarding whether points belong to the scene foreground or background. The maximization of this fitness function through Particle Swarm Optimization leads to the adjustment of the foreground detection parameters. Extensive experimental results confirm the effectiveness of the adopted approach.

1. Introduction

As digital cameras become cheaper, multicamera setups or camera networks are becoming commonplace. Calibrated multiview setups are associated with some strong assumptions and their intrinsic/extrinsic calibration is a tedious process. Nevertheless, their ability to reduce occlusion effects and appearance ambiguities leads to more robust performance of computer vision algorithms, a fact that typically outweighs their disadvantages. Several multicamera-based applications such as semi-automated surveillance [8], target tracking [17], 3D video recording [18,23], human motion modeling [4,28] and sports analysis [9] perform object detection, most commonly using some background modeling-based foreground detection method. Thus, such methods constitute important ingredients of modern multiview computer vision systems.

A common drawback of several existing foreground detection methods is that their performance critically depends on several parameters that require considerable expertise in order to be adjusted properly. Unfortunately, there is no universal parameter set that can generalize optimally across the different conditions that may be encountered. In the typical case, different scenarios that exhibit variable degree of occlusions (e.g., crowded scenes), stopped targets, clutter motion (e.g., flowing water) and global or local illumination changes, require different tuning of the algorithm towards high quality results. Despite its great importance, proper parameter tuning is often overlooked resulting in suboptimal foreground detection output. The need for adaptive parameter tuning is even more pronounced when dealing with online, real-time applications that capture endless video streams (e.g., automated surveillance) where the environmental and other conditions might change considerably over time.

One of the few approaches that deal with this problem is the one adopted by White and Shaw [27], which presents a method that optimizes background subtraction with respect to given ground truth. More specifically, the goal is to optimize two basic parameters of a background subtraction algorithm [24] that is applied to an image sequence acquired by a single camera. The required ground truth consists of manually defined foreground silhouettes. The F measure [22] between the silhouettes calculated by the background subtraction algorithm and the ground truth silhouettes constitutes the fitness function of a given parameter set. Finally, *Particle Swarm Optimization* (PSO) is employed to maximize this fitness function by searching over the space of possible background subtraction parameters.

In this work, we propose a novel method for automatically tuning the foreground detection parameters, utilizing information taken by a multicamera setup. In contrast to White and Shaw [27], the proposed method does not require user intervention at any point of the process and does not assume the availability of ground truth measurements. Thus, it can be applied to the automatic tuning of foreground detection performed on any system that captures endless video streams where ground truth information is not available. Similar to White and Shaw [27], we employ PSO to optimize a fitness function that is defined over a multidimensional foreground detection parameter space.

* Corresponding author. Address: N. Plastira 100, Vassilika Vouton, GR-700-13 Heraklion, Crete, Greece. Fax: +30 2810 391609.
E-mail address: argyros@ics.forth.gr (A. Argyros).

1077-3142/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.cviu.2010.09.003

Instead of using ground truth silhouette images, we employ *confidence maps* that are calculated through the fusion of the foreground images estimated by the multicamera setup. At each step, one such map is produced for every camera of the configuration. Each confidence map consists of scores that represent the cumulative confidence in the multicamera setup regarding whether a pixel belongs to the foreground or not. For each and every camera, the fitness function measures the similarity of the foreground estimate to the confidence map. The fundamental idea behind the definition of the fitness function is that if several cameras agree that a certain point in the scene belongs to the foreground, then this is likely to be so. False positives and false negatives may exist in the process. Nevertheless, it is very unlikely that a consensus will be build around them. As in [27], PSO is used to maximize the fitness function. PSO suggests foreground detection parameters that produce new confidence maps which, in turn, suggest new parameters. The termination of this iterative process provides the parameter vector found to achieve the greatest fitness. Through a series of experiments, we show that both the defined fitness function and optimization process are very suitable for effectively solving the problem of unsupervised adjustment of foreground detection parameters.

The main contributions of this work are (1) the definition of multicamera consensus and the resulting confidence maps in the optimization of the foreground detection parameters, (2) the unsupervised solution of the problem of parameter tuning as opposed to the previous supervised methods requiring ground truth information, and (3) a thorough experimental study of the behavior of the proposed approach with a detailed investigation of various factors that may affect its performance.

The remainder of this paper is organized as follows. In Section 2 the foreground detection algorithm that is used throughout this work is presented. It has to be noted that the selection of the particular method is based on its popularity and performance [3]. Nevertheless, the proposed method can, in principle, be applied to any other background subtraction/foreground detection method. Section 3 defines the confidence maps that guide the optimization process. Section 4 presents the employed optimization algorithm. Section 5 provides a detailed description of the proposed algorithm. Experiments and results are presented in Section 6. Finally, a brief summary and conclusions is given in Section 7.

2. Background modeling and foreground detection

Background modeling and foreground detection is a way to detect moving objects in views acquired by static cameras. The great importance of such methods has given rise to several approaches. According to Piccardi [21], such methods typically operate at the pixel level. The simplest ones directly subtract the average, median or running average of a number of frames from the current view. Other methods use kernel density estimators and mean-shift based estimation [10,12]. In [20], the notion of eigen-background is defined.

One of the best performing methods is the one proposed by Stauffer and Grimson [24] that models the appearance of each image pixel as a mixture of Gaussians. Because of its effectiveness and popularity [3], our work considers this method as the basis of the proposed, unsupervised parameter optimization approach. More specifically, we employ the variant proposed by Zivkovic [29]. For the sake of self completeness, an introduction to this method is provided.

Given a sequence of images, let $\vec{x}^{(t)}$ be a pixel of image $I^{(t)}$ at time t in some colorspace (i.e., RGB). The background model is estimated from a training set $X_T = \{x^{(t)}, \ldots, x^{(t-T)}\}$ where T determines the time period for which the model's history is extended. Each pixel is modeled as a M component Gaussian Mixture Model (GMM) given by

$$\hat{p}(\vec{x}|X_T, fb) = \sum_{m=1}^{M} \hat{\pi}_m N(\vec{x}; \hat{\vec{\mu}}_m, \hat{\sigma}_m^2 I), \tag{1}$$

where $\hat{\vec{\mu}}_1, \ldots, \hat{\vec{\mu}}_M$ are the estimates of the means and $\hat{\sigma}_1, \ldots, \hat{\sigma}_M$ are the estimates of the variances of the GMM components. *fb* denotes the fact that the recent history contains observed values belonging to both the foreground (*f*) and the background (*b*). Given a new data sample $\vec{x}^{(t)}$ at time t, the recursive update equations of mixing weights, means and variances are:

$$\hat{\pi}_m \leftarrow \hat{\pi}_m + \alpha\left(o_m^{(t)} - \hat{\pi}_m\right) - \alpha c_T, \tag{2}$$

$$\hat{\vec{\mu}}_m \leftarrow \hat{\vec{\mu}}_m + o_m^{(t)}(\alpha/\hat{\pi}_m)\vec{\delta}_m, \tag{3}$$

$$\hat{\sigma}_m^2 \leftarrow \hat{\sigma}_m^2 + o_m^{(t)}(\alpha/\hat{\pi}_m)\left(\vec{\delta}_m^T\vec{\delta}_m - \hat{\sigma}_m^2\right), \tag{4}$$

where $\vec{\delta}_m = \vec{x}^{(t)} - \hat{\vec{\mu}}_m$, α is the constant that represents an exponentially decaying envelope utilized to attenuate the effect of past data and c_T a small bias factor, typically set to 0.01 (see Zivkovic [29] for details). A sample is close to a GMM component if its Mahalanobis distance from the mode is smaller than a certain threshold, typically set equal to three standard deviations. Based on this, the ownership $o_m^{(t)}$ for a newly arrived sample is set to 1 for the GMM component with the larger mixing weight among all the components that their distances from the sample is less than the predefined threshold and 0, otherwise. The squared distance from the mth component is computed by $D_m^2(\vec{x}^{(t)}) = \vec{\delta}_m^T\vec{\delta}_m/\hat{\sigma}_m^2$. Updates of π_ms must be followed by a normalization so that they add up to one.

Background modeling starts with one GMM component centered on the first sample. While the new samples that arrive are not within three standard deviations from the existing modes of the GMM, new components are generated with $\hat{\pi}_{M+1} = \alpha$, $\hat{\vec{\mu}}_{M+1} = \vec{x}^{(t)}$ and $\hat{\sigma}_{M+1} = \sigma_0$ where σ_0 is the initial variance. During updates, if a mixing weight $\widehat{\pi_m}$ becomes negative, the corresponding mixture component is removed from the GMM and the mixing weights of the remaining components are normalized to sum to one. Moreover, if a newly imported component forces the total number of components to increase beyond a certain threshold, the component with the smallest mixing weight assigned to it is excluded from the mixture.

Given that the components of the mixture are sorted in a descending order of their mixing weights, it is assumed that the background can be modeled by the set B of the largest GMM components as:

$$p(\vec{x}|X_T, fb) \sim \sum_{m=1}^{B} \hat{\pi}_m N\left(\vec{x}; \hat{\vec{\mu}}_m, \sigma_m^2 I\right), \tag{5}$$

where

$$B = \arg\min_j \left\{ j | \sum_{m=1}^{j} \hat{\pi}_m > (1 - c_f) \right\} \tag{6}$$

and c_f is the maximum allowable sum of mixing weights of the GMM components modeling the foreground.

Following the above analysis, an observed pixel is part of the background, if it is found to be close to one of these B Gaussian components. Otherwise, this pixel is assigned the foreground label. An example outcome of this foreground detection method is shown in Fig. 1.

3. Multiview camera setup

The foreground detection method presented in Section 2 operates on an image sequence acquired by a single, static camera.

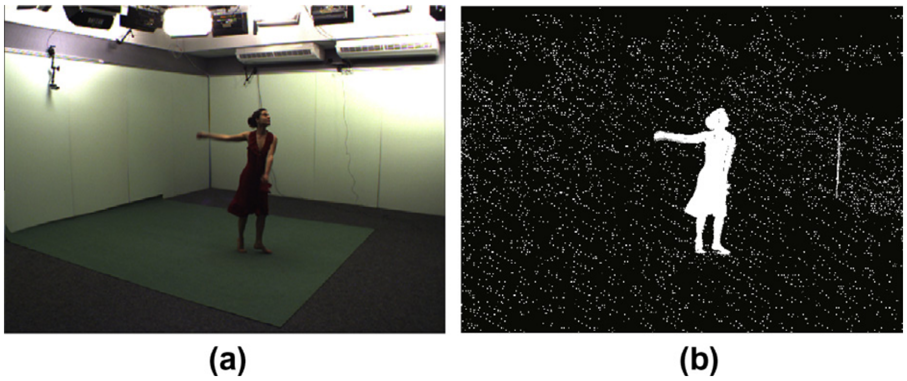

Fig. 1. (a) A frame of Inria's Dancer sequence and (b) foreground detection output. White and black pixels correspond to foreground and background, respectively.

The straightforward approach to performing foreground detection in a multicamera setup is to employ it independently in each of the acquired views. A basic idea behind this work is that the joint observation of a given 3D space by a set of cameras can be used to provide information that may guide the joint optimization of the foreground detection parameters. A given observed 3D point, either belongs to the scene foreground or scene background. Thus, the *visual hull* [15] estimated through *volume intersection* [16] can be used to compute the *multiview configuration consensus* regarding the foreground of a given scene.

More specifically, each camera of the configuration votes in a common voxel space for occupied voxels by projecting an estimation of its own foreground image on this space. The voxel space describes a discretization of the actual space. A voxel can be considered, by a single view, as being occupied by some object or not. This occupancy information is all that is required to calculate the multicamera consensus regarding the objects present in the scene. After the occupancies are calculated, the voxel space can be back-projected to every view to calculate a set of *confidence maps*, one per view. The use of voxel occupancies as a way to combine information from multiple views has been proposed at [11] where a probabilistic framework for fusing silhouette cues is presented.

What follows, is a detailed presentation of how the multicamera consensus and the individual confidence maps are built.

3.1. Multicamera consensus

To calculate the multicamera consensus, a 3D voxel space of the actual scene is defined. This space is sampled to create a 3D grid, $G = \{G^0, G^1, \ldots, G^n\}$ where each $G^c = (X_c, Y_c, Z_c)$ is a 3D point. General perspective projection of a 3D point $(X_c, Y_c, Z_c, 1)$ to a 2D point (x_c, y_c, f_c) on the ith view plane can be calculated given the corresponding projection matrix $P_i = C_i[R_i|T_i]$ through

$$(x_c, y_c, f_c)^T = C_i[R_i|T_i](X_c, Y_c, Z_c, 1)^T, \tag{7}$$

where C_i is the camera calibration matrix, R_i the rotation matrix and T_i the translation vector with respect to a world-centered coordinate system. In the general case, the cameras of a multiview configuration cannot be fully aligned on a common field of view (FOV), so a number of 3D points will fall outside the FOV of some cameras. For the view plane of camera i with dimensions $w_i \times h_i$ we define the function $L_i(x,y)$ that labels the projections falling inside the camera FOV as

$$L_i(x,y) = \begin{cases} 1 & 1 \leqslant x \leqslant w_i \wedge 1 \leqslant y \leqslant h_i, \\ 0 & \text{otherwise}. \end{cases} \tag{8}$$

Furthermore, we denote by S_i the silhouette image (as the one shown in Fig. 1b) taken from camera i, where $S_i(x,y) = 1$ for foreground pixels and $S_i(x,y) = 0$ for background pixels. Occupancy scores $O(X_k, Y_k, Z_k)$ of 3D points of G are computed as

$$O(X_k, Y_k, Z_k) = \begin{cases} 1 & s = l > \frac{|C|}{2} \\ 0 & \text{otherwise} \end{cases}, \quad \forall k \in [0, n]. \tag{9}$$

In Eq. (9), $|C|$ is the number of cameras used. l is termed the *visibility factor* (see Fig. 2a) and s the *intersection factor* (see Fig. 2b). These factors are defined as

$$l = \sum_{i \in C} L_i\left(\frac{x_k^i}{f_k^i}, \frac{y_k^i}{f_k^i}\right), \quad s = \sum_{i \in C} S_i\left(\frac{x_k^i}{f_k^i}, \frac{y_k^i}{f_k^i}\right), \tag{10}$$

where $(x_k^i/f_k^i, y_k^i/f_k^i)$ are the projections of (X_k, Y_k, Z_k) at view plane i.

3.2. Confidence maps

Confidence maps $\mathscr{C}_i(x,y)$ are computed for every view i by accumulating the occupancy scores of the back-projections of the view planes on every slice of the grid G. Slices are considered to be 3D point sets of fixed Z_c, with Z_c taking discrete values in the range of $[Z_{min}, Z_{max}]$. Therefore, confidence maps are calculated through:

$$\mathscr{C}_i(x,y) = \sum_{Z_{min} \leqslant z \leqslant Z_{max}} O(X', Y', z) \tag{11}$$

for every (x,y) such that $1 \leqslant x \leqslant w_i \wedge 1 \leqslant y \leqslant h_i$. Given the 3×4 projection matrix $P_i = [p_{mn}^i]$ of view i the projections X' and Y' of x and y are calculated analytically as

$$Y' = \frac{z(xp_{23}^i - yp_{13}^i + m(yp_{33}^i - p_{23}^i))}{yp_{12}^i - xp_{22}^i - m(yp_{32}^i - p_{22}^i)} + \frac{m(yp_{34}^i - p_{24}^i) + xp_{24}^i - yp_{14}^i}{yp_{12}^i - xp_{22}^i - m(yp_{32}^i - p_{22}^i)} \tag{12}$$

and

$$X' = \frac{Y'(yp_{32}^i - p_{22}^i) + z(yp_{33}^i - p_{23}^i) + yp_{34}^i - p_{24}^i}{p_{21}^i - yp_{31}^i}, \tag{13}$$

where

$$m = \frac{xp_{21}^i - yp_{11}^i}{p_{21}^i - yp_{31}^i}. \tag{14}$$

After their calculation, the values of the confidence maps are normalized to the range [0, 1]. The closer a value is to 1, the higher the estimated confidence that the corresponding pixel belongs to a foreground object.

Fig. 3 shows examples of computed confidence maps. As can be verified, confidence maps attenuate the holes in the silhouettes but also the noise in the background. The intuition behind this result is that although false positives and false negatives may exist in indi-

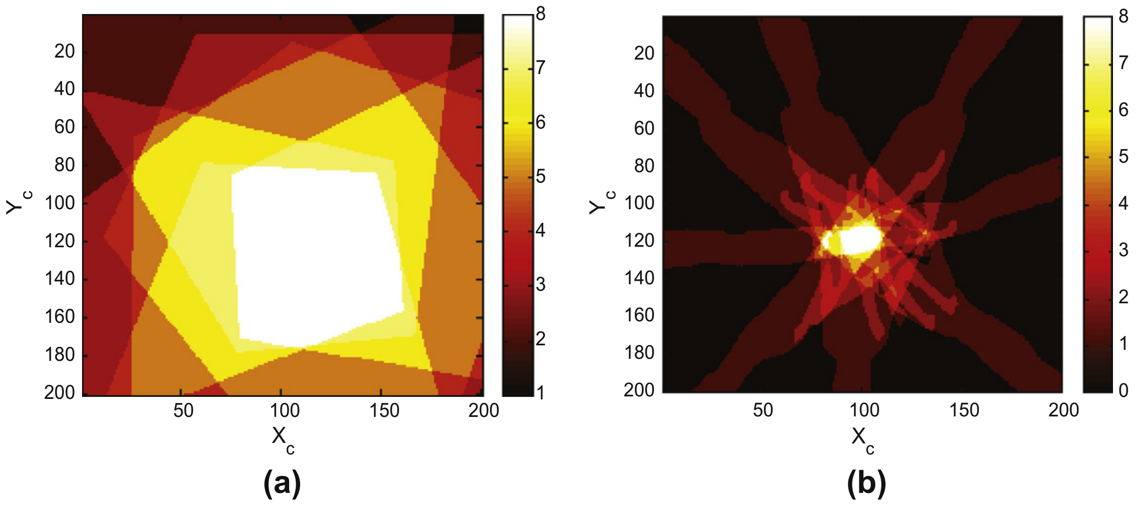

Fig. 2. (a) A slice of the grid G for Z_c = 0 cm. Different gray level values denote scene regions of variable visibility from a multiview configuration of eight cameras. Dark regions are visible from one view while bright regions are visible from all the cameras. (b) A slice of the grid G is shown (for Z_c = 100 cm). Each view projects on this slice its captured silhouette. White areas correspond to silhouette intersections from all the views of the configuration.

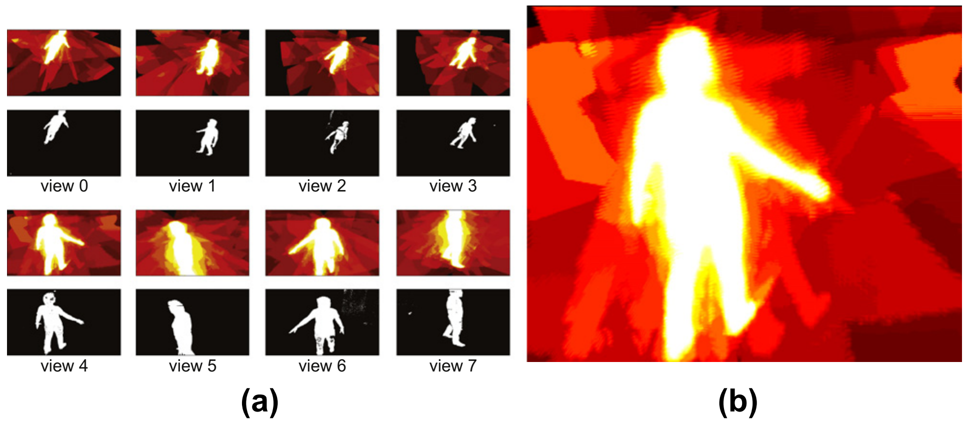

Fig. 3. (a) The confidence maps and silhouette images for a single frame across all views of an 8-camera configuration. In (b), the confidence map obtained in view 4 is shown in greater detail. Brighter colors correspond to higher confidence values.

vidual camera foreground detections, it is very unlikely that a strong consensus is built around them. Thus, confidence maps represent more robustly the segmentation of a scene into foreground and background, compared to the single view silhouette estimates.

4. Particle Swarm Optimization

Classical approaches on solving optimization problems are often based on the evaluation of the derivatives of the defined objective function. In real-world optimization problems, the analytical expression of the objective function is not known or it is multimodal, i.e., has several local minima. Additionally, its derivatives may not-even be defined at certain points of the parameter space. To cope with such problems, derivative-free optimization algorithms have been proposed. One such approach is Particle Swarm Optimization (PSO) [14]. PSO is a population based stochastic optimization method that utilizes swarm intelligence to find extrema of nonlinear continuous functions (a.k.a. *objective* or *fitness* functions). It is similar to other evolutionary techniques like *Genetic Algorithms* [13] with the major difference of having no crossover and mutation operators. PSO exhibits better performance compared to several other optimization methods [1] and is very efficient in terms of computational cost.

Particle Swarm Optimization is an attractive optimization method for the problem at hand for several reasons. It performs well with non-smooth, multimodal objective functions and requires a relatively low number of objective function evaluations [1]. It depends on a very few parameters and it scales well with the number of parameters to be optimized. Finally, it is inherently parallel, leaving room for parallel implementations that can drastically reduce the computation time required for optimization, especially when this is intended to be performed on-line.

4.1. Social optimization

PSO is based on social interactions between the atoms of a population in order to optimize a problem modeled with a specific fitness function. The method is inspired by the social behavior exhibited in flocks of birds and schools of fishes. As such, it handles populations of particles that are defined in the optimization space. A social network between individuals (i.e., particles) is defined. The particles are candidate solutions that are initialized randomly. The social network determines the interactions that can take place (e.g., particles can only interact with their neighbors). During the execution of the PSO algorithm, particles evaluate the fitness of the candidate solutions that represent and store in memory the parameters achieving the optimum fitness values. Moreover, they adjust their velocities through predefined update equations. Finally, they move in the parameter space, i.e., update their positions according to a random linear blending performed upon two velocity vectors. One of these vectors points towards the particle's local best solution and the other towards the best solution in a neighborhood of particles. This process evolves iteratively, where each iteration is called a *generation*, until a termination criterion is met. Such criteria include the convergence of the whole or of a portion of the particle population to a single solution, the execution of an upper bound of iterations, the achievement of a specific fitness score, etc.

A great number of PSO variants have been proposed. In this work, the simplest form of the PSO algorithm, called *canonical* PSO [7] has been employed. Other popular variants include the *fully informed* PSO [19] as well as variants that define dynamic neighborhood topologies [25] and those that utilize enhanced diversity at updating [2]. Variants have also been defined by using heuristic velocity update rules or by explicitly handling discrete optimization problems [6].

4.2. Canonical PSO

In canonical PSO, the topology of the population reduces to only one neighborhood. Following the notation introduced in [27], every particle holds its current position (current candidate solution, set of parameters) in a vector x_t and its current velocity in a vector υ_t. Moreover, each particle stores in vector p_i the position at which it achieved, up to the current generation t, the highest fitness score. Finally, the swarm as a whole, stores in vector p_g the best position encountered across all particles of the swarm. p_g is broadcasted to the entire swarm, so every particle is aware of the current global optimum. The update equations that are applied in every generation t to reestimate the particle velocities and positions are

$$\upsilon_t = K(\upsilon_{t-1} + c_1 r_1 (p_i - x_{t-1}) + c_2 r_2 (p_g - x_{t-1})) \tag{15}$$

and

$$x_t = x_{t-1} + \upsilon_t, \tag{16}$$

where K is a constant *constriction factor* [5] defined as

$$K = \frac{2}{\left|2 - \psi - \sqrt{\psi^2 - 4\psi}\right|}, \quad \psi = c_1 + c_2. \tag{17}$$

In Eqs. (15) and (17), c_1 is called the *cognitive component*, c_2 is termed the *social component* and r_1,r_2 are random samples of a uniform distribution in the range [0,1]. Finally, $c_1 + c_2 > 4$ must hold [5]. In all performed experiments the values $c_1 = 2.8$ and $c_2 = 1.3$ were used.

As mentioned earlier, the particles are initialized at random positions and their velocities are initialized to zero. Each dimension of the multidimensional parameter space is bounded in some range. If, during the position update, a velocity component forces the particle to move to a point outside the bounded search space, this component is zeroed and the particle doesn't perform any move at the corresponding dimension.

5. Optimization of foreground detection parameters

The proposed algorithm is an iterative procedure that utilizes canonical PSO to search for the optimal parameter vector across the parameter space of the foreground detection algorithm presented in Section 2. The optimal parameter vector is defined to be the one that maximizes the similarity between silhouettes and confidence maps across all available views. Each particle position corresponds to a set of foreground detection parameter values. During particle evaluation, a foreground detection instance is initialized using the particle's position and applied to an image subsequence to produce a set of silhouette estimates. The use of sequences instead of single frames is mandatory because, by definition, the foreground detection algorithm requires a history of observations in order to produce reliable results.

The proposed iterative optimization process consists of the following steps (a) calculation of the confidence maps based on the current silhouette estimates, (b) optimization of the foreground segmentation parameters using the computed confidence maps, and (c) calculation of new silhouette estimates using the optimized parameters. By iterating the above steps in a closed loop, both the estimated parameters and the quality of the produced silhouettes get improved. A similar idea in the field of Machine Learning is employed in the principle of *generalized policy iteration* [26]. The defined fitness function measures the similarity between confidence maps and silhouette estimates across a given image sequence and for every view. Silhouette estimates are computed as reported in Section 2 from an instance of the foreground detection algorithm that is initialized by the position vector of a given particle. Confidence maps are produced by the silhouette estimates and the additional calibration information of the multiview configuration, as detailed in Section 3.2.

More specifically, let $S_i^t(x,y)$ denote a point of the silhouette image of frame t captured by camera i. Let also $\mathscr{C}_i^t(x,y)$ denote the value of the confidence map for the same point. The distance $D_{A,i,t}$ between silhouettes and confidence maps for a set of points A is calculated as:

$$D_{A,i,t} = \sum_{(x_p,y_p)\in A} \left|S_i^t(x_p,y_p) - \mathscr{C}_i^t(x_p,y_p)\right|. \tag{18}$$

If we denote with $P_{fg,i}^t$ the set of silhouette pixels of frame t of view i (i.e., foreground pixels) and with $P_{bg,i}^t$ the set of the background pixels, then the fitness function is defined as

$$F = \sum_{t\in[0,T]} e^{(1-r_t/2|C|)}, \tag{19}$$

where

$$r_t = \sum_{j\in C} \left(\frac{D_{P_{fg,j}^t,j,t}}{\left|P_{fg,j}^t\right|} + \frac{D_{P_{bg,j}^t,j,t}}{\left|P_{bg,j}^t\right|} \right). \tag{20}$$

Algorithm 1 provides a summary of the computation of the fitness function while Algorithm 2 provides a summary of the full optimization process.

Algorithm 1. Computation of the fitness function

```
Input: Particle $\mathscr{P}, T, N_c$
Output: Fitness score $F$
$F = 0$
foreach $l = 1,2,\ldots,T$ do
  Compute silhouettes $S_i, \forall i \in [1,N_c]$ (as described in Sec. 2);
  Compute confidence maps $\mathscr{C}_i, \forall i \in [1,N_c]$ (Eq. (11));
  Compute $P_{fg,i,l}$, $P_{bg,i,l}$, $\forall i \in [1,N_c]$;
  $r_l = \sum_{i\in[1,N_c]} \left(\frac{D_{P_{fg,i,l}}}{|P_{fg,i,l}|} + \frac{D_{P_{bg,i,l}}}{|P_{bg,i,l}|}\right)$ (Eq. (20));
  $F = F + e^{(1-r_l/2|C|)}$;
return (F);
```

6. Experiments

The goal of the performed experiments is (a) to show whether the proposed method can be applied successfully to image sequences acquired by a calibrated multiview configuration in order to automatically tune the foreground detection parameters and produce optimal silhouette images in a totally unsupervised manner and (b) to investigate the influence of several factors (i.e., PSO parameters, noise level, camera number and topology, etc.) on the quality of the obtained results.

6.1. Parameter selection

The performance of foreground detection is governed by the learning rate parameter α (Eqs. (2)–(4)) that determines the speed of the adaptation. A uniform update speed is enforced by setting $\alpha = 1/T$.

Algorithm 2. Optimization of the foreground detection parameters.

```
Input: Number of PSO generations $N_g$, length of frame
  sequence $T$, PSO population size $N_p$, number of cameras $N_c$
Outpur: Optimal foreground detection parameter vector $\overline{\mathscr{P}^*}$
$F_{max} = 0$;
Initialize $N_c \times N_p$ particles $p_i$ randomly (random $x_i, v_i = 0$);
foreach $n = 1,2,\ldots,N_g$ do
    Perform particle $p_i$ flight, $\forall i \in [1, N_c \times N_p]$ (Eq. (15));
    Compute fitness $F_i$ of $p_i, \forall i \in [1, N_c \times N_p]$ (through Algorithm 1);
    if $F_i > F_{max}$ then
        $F_{max} = F_i$;
        $\overline{\mathscr{P}^*} = p_i$;
    Update velocity of $p_i$, $\forall i \in [1, N_c \times N_p]$ (Eq. (16));
return ($\overline{\mathscr{P}^*}$)
```

Another important parameter is the threshold T_b on the squared Mahalanobis distance upon which it is decided if a given sample is close to a background GMM component or not. It must be noted that T_b is different from the threshold T_g that specifies whether a sample belongs to any of the mixture components modeling either background or foreground. According to Zivkovic [29], typical values are $T_b = 16\sigma_m^{(t)}$ and $T_g = 3\sigma_m^{(t)}$ for the mth component at time step t.

In general, it is proposed that a total of four Gaussian components are sufficient for the purposes of foreground detection. Therefore, in our experiments this parameter did not vary. Moreover, let $T_b = 1 - c_f$ where it holds that $0 \leqslant T_b \leqslant 1$. The threshold T_b determines (see Eqs. (5) and (6)) the number of mixture components that model the background. In order for the background modeling to be valid, T_b must have a value that allows for the background to be modeled by at least one Gaussian component. Typically, $c_f = 0.1$ which leads to $T_b = 0.9$. Finally, the initial variance σ_0 of the newly imported components in the mixture, influences the speed of adaptation. A typical value for this parameter is $\sigma_0 = 10$.

As the parameters $\{\alpha, T_b, T_g, \sigma_0\}$ have a great impact on the final result of the foreground detection process, they were selected as the target variables of the proposed optimization process.

6.2. Experimental setup

The experimental validation of the proposed method was based on two datasets. The first is the "Dancer" dataset[1] of Inria's 4D repository. This dataset captures the movements of a female dancer through a configuration of eight calibrated cameras. Each view captures 251 synchronized frames of size 780×582. The first 50 frames contain only scene background and are provided for proper initialization of the background modeling process. From those, 49 frames were omitted, so the resulting sequence starts with a single frame showing the scene background in isolation. On top of the actual data, the dataset comes with a set of preprocessed silhouettes (one per frame). These data are not used in the optimization process but form a basis for the quantitative evaluation of the non-supervised foreground detection algorithm.

The second dataset[2] is a synthetic, noise-free dataset, showing a 3D rendered model of a Kung-Fu girl in action. This has been acquired by a virtual multiview setup of 25 cameras and contains 201 synchronized frames of image size 320×240. There is a single frame showing the scene background in isolation. This dataset also comes with a ground truth set of silhouettes that is produced automatically by rendering the 3D model with no lights, resulting in a white silhouette on a black background.

The description of the foreground detection method in [29] suggests a parameter set that performs relatively well in the general case. We refer to these parameters as *typical parameters*. Throughout our experiments, we evaluate the typical parameters and the parameters suggested by our methodology against the available ground truth. This evaluation involves a comparison of the silhouette images produced by a set of parameters against the available ground truth silhouettes. More specifically, let Ω be the set of all image pixels for all cameras and time instances. Then the measure used for comparing the resulting silhouette images to ground truth is:

$$q = 1 - \frac{D_\Omega}{|\Omega|}, \tag{21}$$

where

$$D_\Omega = \sum_{t \in [0,T]} \sum_{(x,y) \in \Omega} |S^t(x,y) - T^t(x,y)| \tag{22}$$

and $T^t(x,y)$ denotes the ground truth available for point (x,y) at time t. A value of $q = 1$ signifies silhouette images identical to the ground truth and, therefore, perfect foreground detection parameters.

6.3. Dancer dataset

We present quantitative and qualitative results obtained from the application of the proposed method on the dancer dataset. As detailed in Section 6.1, the most critical foreground detection parameters are α, T_b, T_g and σ_0. In a first experiment, we used a population of 15 particles running PSO for 50 generations on the 4D parameter space $\{\alpha, T_b, T_g, \sigma_0\}$. Each particle is evaluated on the entire dancer sequence. We call this the *exhaustive* or the *all-frames experiment*.

In a second experiment, the self-evaluation of each particle considered only the first 10 frames of the entire sequence. We call this experiment the *10-frames experiment*. In this case, a population of eight particles run PSO for 20 generations on the 3D parameter space of $\{T_b, T_g, \sigma_0\}$. The reason for excluding parameter α is that a value of α that is optimal on the small, 10-frames time window, cannot generalize well in a sequence of extended length. Therefore, α was fixed to the typical value while PSO was set to jointly optimize parameters T_b, T_g and σ_0.

The parameter vectors estimated in the two experiments were evaluated on the entire sequence. The typical parameter vector as well as the initialization parameter vector of the second experiment were also evaluated. These four parameter vectors are listed in Table 1. Table 1 also reports the mean fitness values across the whole frame range of the sequence achieved by each parameter

[1] Available for download at http://charibdis.inrialpes.fr/public/viewgroup/1.

[2] Available for download at http://www.mpi-inf.mpg.de/departments/irg3/kungfu/.

Table 1
Evaluation of foreground detection parameters on the dancer sequence.

Parameter set	T_b	T_g	σ_0	α	Mean fitness value
Typical	16.0	9.0	11.0	0.0001	2283.29
Initialization	3.3	13.2	2.7	0.0001	211.71
10-frames best	18.6	19.0	37.0	0.0001	6389.85
All-frames best	11.9	16.7	41.6	0.0004	**6613.91**

set. The detailed fitness graph for all parameter sets in all sequence frames is shown in Fig. 4.

From these results, it can be verified that the parameter set returned from the exhaustive experiment was the best, followed by the parameter set resulting from the 10-frames experiment. Those two sets achieved far better fitness scores than the typical parameters, having a marginal difference between them. Fig. 5 shows how those fitness scores translate to ground truth similarity.

Two important conclusions can be drawn from these results. First, there is a consistency between fitness function scores and ground truth similarity scores, thus the fitness function is well defined. Second, the results of the exhaustive experiment are very similar to the results of the 10-frames experiment, leading to the conclusion that the parameters found on the small training set generalize very well for the rest of the sequence assuming that there are no major changes in the environment. This is an important observation that can be exploited to avoid the significant additional computational overhead of optimizing a large population of particles across many generations on the whole sequence at a small quality pay-off. This also demonstrates that the proposed method can be used for the automatic tuning of parameters on streaming sequences using just a small number of frames to estimate the proper parameters. Examples of the silhouette images produced by applying the four different instances of foreground detection on a specific view and frame together with the ground truth are shown in Fig. 6. As it can be verified, the noise patterns appearing in the images corresponding to the initialization and typical parameter sets are missing from the image corresponding to the optimal parameter set.

We furthermore isolated the particle that returned the optimal position for the 10-frames experiment and we recorded its route to this solution. The fitness of this particle as a function of generations is illustrated in Fig. 7. The plot indicates that the proposed method requires approximately 15 generations to optimize the parameters.

6.4. Kung-Fu girl dataset

We also conducted the 10-frames experiment on the Kung-Fu girl dataset (8 particles, 20 generations, training set of 10 frames, $\{T_b, T_g, \sigma_0\}$ parameter space). Following the same approach as in the case of the dancer dataset, we evaluated the three parameter vectors shown in Table 2. The corresponding fitness graphs are shown in Fig. 8a. Similarity to ground truth was computed as shown in Fig. 8b. Finally, examples of silhouette images from the three detection instances that correspond to the parameter sets of the experiments are shown in Fig. 9.

6.5. Noise effects

The presence of noise in the input image is responsible for increasing the number of detected foreground pixels. This is because color variations due to noise are more likely to manifest themselves as foreground rather than as a significantly varying background. This can be observed on the output of the typical parameters for the dancer dataset (Fig. 6) where foreground pixels are distributed, following a certain camera dependent noise pattern, across the entire image area. Thus, in the dancer sequence experiments, the proposed optimization seeks the optimal parameter set that also compensates for image noise. In the case of the experiment with the synthetic, noise-free Kung-Fu girl dataset, the algorithm just optimizes the similarity to the ground truth.

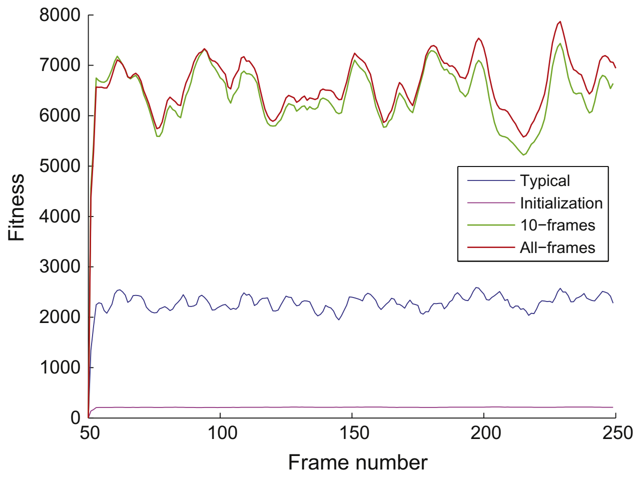

Fig. 4. Fitness curves of the four parameter vectors for the dancer dataset experiments.

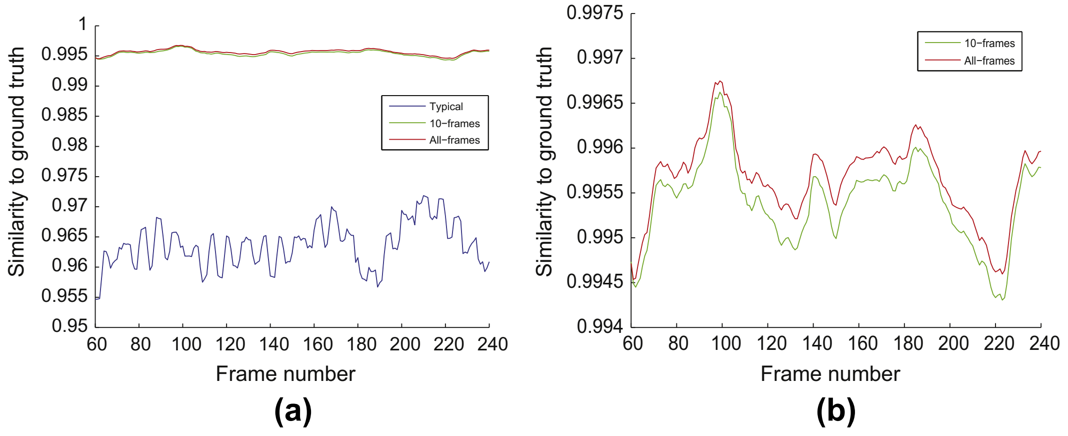

Fig. 5. (a) Comparison of the silhouettes produced by each parameter vector to the ground truth, (b) the performance of the 10-frames and exhaustive experiments, isolated.

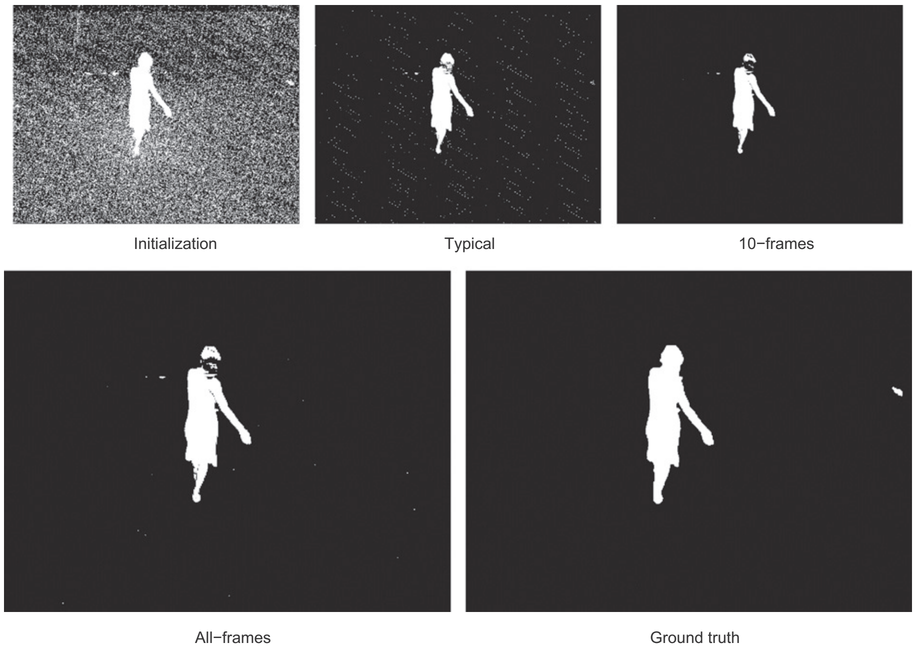

Fig. 6. Example silhouettes calculated by the foreground detection algorithm for frame #110 of the dancer sequence as shown from camera #3 and for the four different parameter sets. The ground truth silhouette is also provided as a reference.

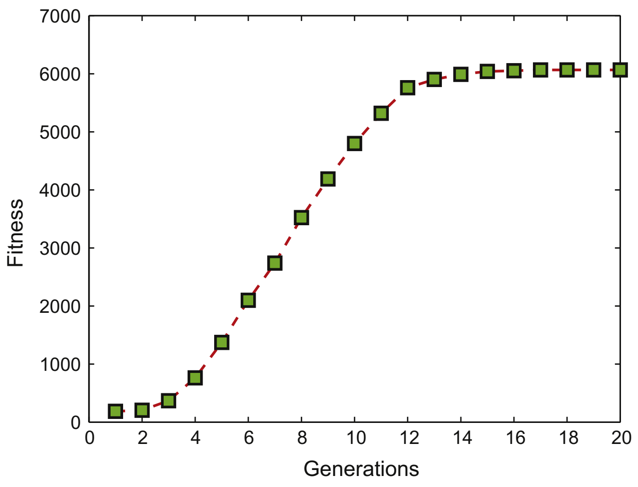

Fig. 7. The route of the initialization parameter set to the optimal position. Markers are placed at the fitness scores that the particle achieved at each generation.

Table 2
Evaluation of foreground detection parameters on the Kung-Fu girl dataset.

Parameter	T_b	T_g	σ_0	Mean fitness value
Typical	16.00	9.00	11.00	4322.30
Initialization	39.50	14.20	27.40	805.99
10-frames best	3.05	2.60	1.30	**10489.90**

As it is shown in Fig. 9, the silhouette image produced with the best parameter set is almost identical to the ground truth, without any holes. On the contrary, the corresponding result for the dancer sequence contains some holes, as a result of the presence of noise.

A series of experiments were conducted to systematically measure the behavior of the proposed algorithm to various noise levels. In these experiments we contaminated the original Kung-Fu girl dataset with three different levels of Gaussian noise ($\mu_1 = 0, \sigma_1^2 = 0.0001$), ($\mu_2 = 0, \sigma_2^2 = 0.00025$), ($\mu_3 = 0, \sigma_3^2 = 0.0005$). Next, we conducted the 10-frames experiment on the resulting datasets and evaluated the results. Finally, we compared the results against the typical parameters. Mean fitness values for the various noise levels are shown in Table 3. Fitness and similarity graphs are shown in Fig. 10, while silhouette examples with the corresponding ground truth are shown in Fig. 11.

As it can be verified, although parameter optimization is affected by noise, in all three cases the suggested parameters result in silhouettes closer to the ground truth than those produced by the typical parameter set. Moreover, for the case where $\sigma_1^2 = 0.0001$, we found that the typical parameters were very close to the optimal parameters returned by the optimization procedure (see Table 4). This might serve as an indication that the typical parameters are tuned to deal with this particular level of image noise. Another interesting observation is that as the noise level increases, the optimization method automatically, but also reasonably, increases the parameter σ_0.

6.6. Camera placement and number of cameras

Another interesting problem dimension is the variability of the obtained results with respect to the placement of the available cameras and their number. The topology of the camera network highly influences the results. More specifically, the method fails to optimize the foreground detection parameters if the cameras arrangement does not permit the accurate voting in the voxel space. As an example, consider a configuration where cameras are placed in one side of the foreground object, only. The fact that large parts of the foreground object are not visible by any of the cameras results in a voxel space that does not accurately represent the object's 3D structure. This produces inaccurate confidence maps which, in turn, leads the parameter optimization process far from its optimal values.

Provided that the cameras are placed in a way that surrounds the foreground objects, the increase of the number of cameras does not improve considerably the obtained results. In order to examine the effects of the number of cameras on the performance of the

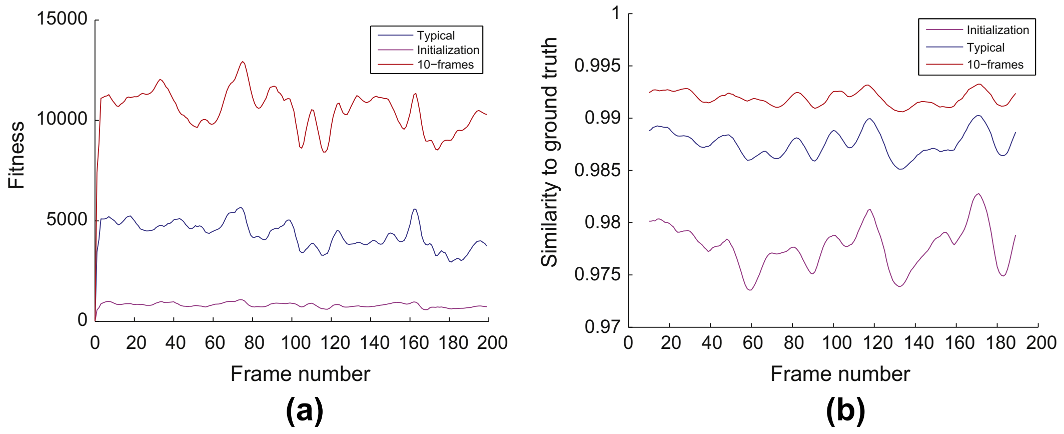

Fig. 8. (a) Fitness curves for the Kung-Fu girl experiments, (b) comparison of the silhouettes produced by each parameter vector to the ground truth for the Kung-Fu girl dataset.

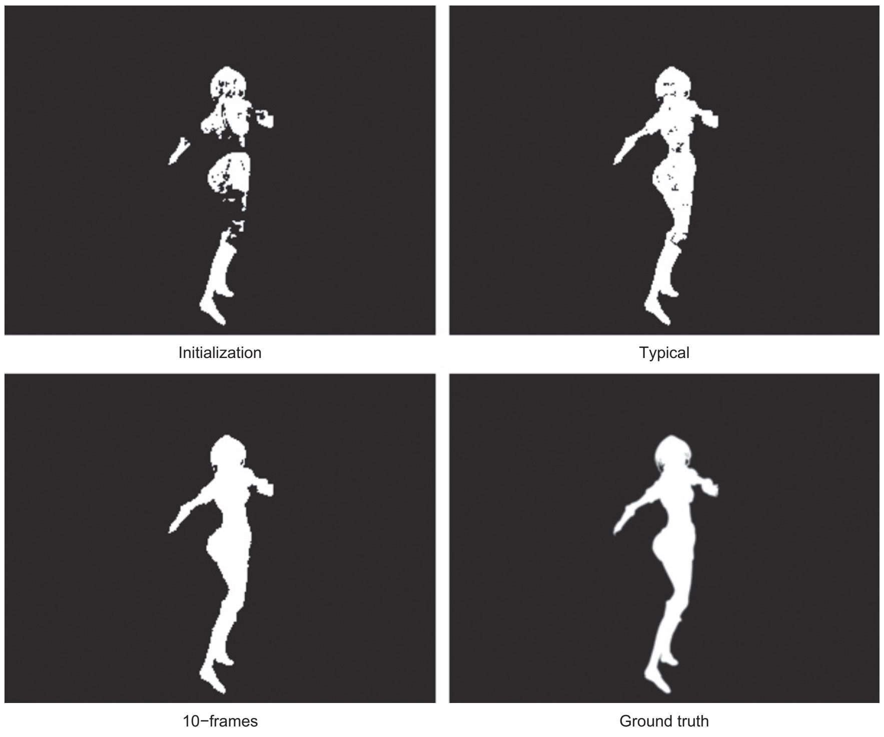

Fig. 9. Silhouettes calculated by the foreground detection algorithm for frame #127 of the Kung-Fu sequence (camera #14) for the three parameter vectors. The ground truth silhouette is also provided.

Table 3
Mean fitness values for various levels of noise contamination of the Kung-Fu girl sequence.

Parameter set	σ_1^2	σ_2^2	σ_3^2
Typical	3661.52	412.25	231.62
10-frames best	**3725.85**	**1473.76**	**859.30**

method, we conducted experiments on the Kung-Fu girl dataset, each time utilizing a different camera subset of the original 25-camera configuration. Sixteen out of the 25 views have nodal points arranged on a circle and optical axes pointing towards the center of this circle. We considered 11 different camera subsets with a number of cameras ranging between 6 and 16. In each case, cameras were distributed as evenly as possible over the entire circle. For all the 11 configurations tested, the resulting fitness value remained practically unchanged and equal to the one reported in the full 25 cameras experiment presented in Section 6.4. Analogous experiments with the dancer data set led to exactly the same performance.

6.7. Optimization of individual camera parameters

In previous experiments a single parameter vector is optimized and used for the entire camera set. This vector defines a low dimensional search space for the optimization algorithm. It is known that the canonical PSO algorithm performs very efficiently in such low dimensional spaces where it only needs to utilize a

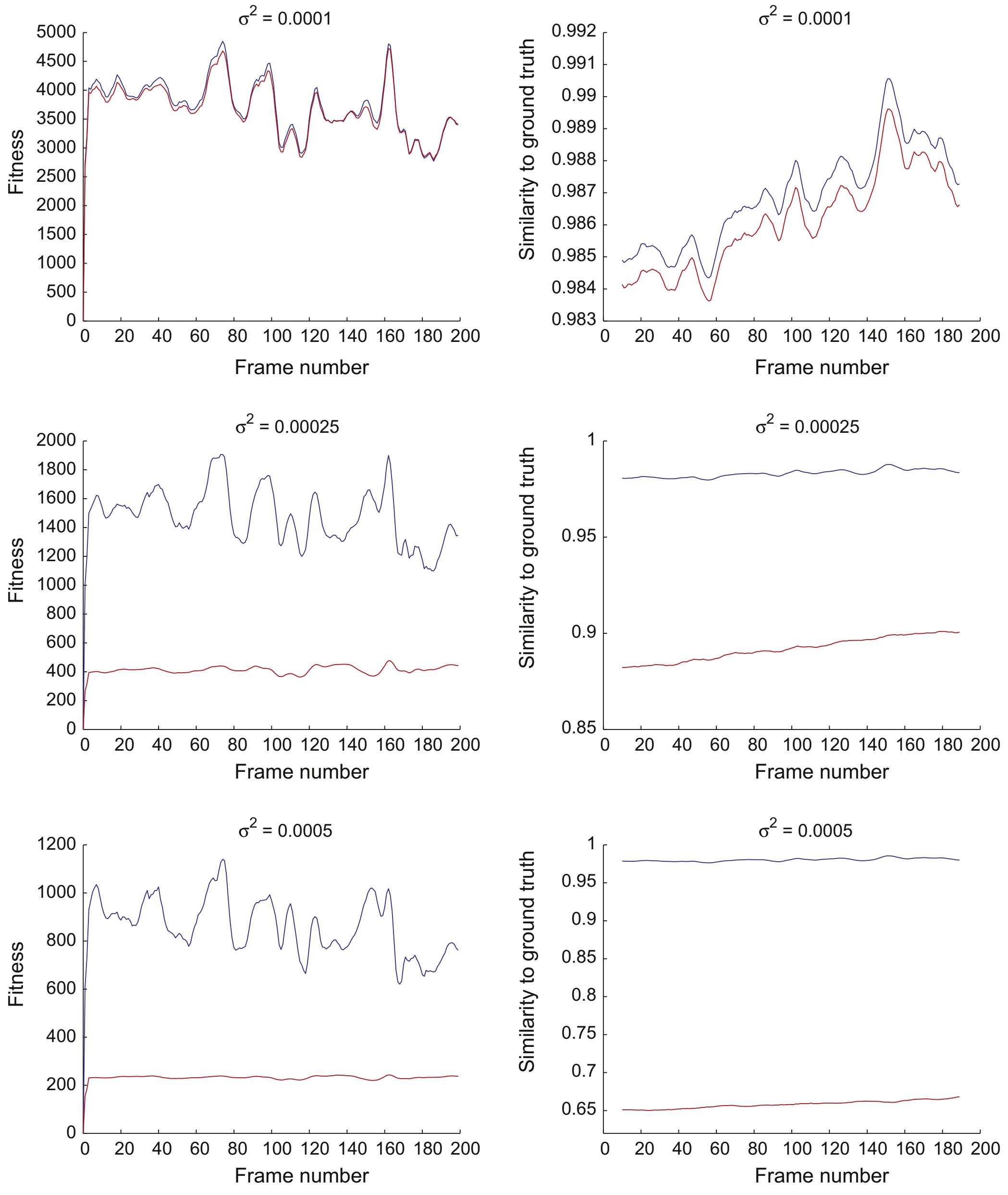

Fig. 10. Fitness graphs for the 10-frames and the typical parameters experiments on the Kung-Fu girl dataset for various noise levels (left) and comparison with the ground truth (right). Blue curves correspond to the 10-frames experiments and red curves to the typical parameters set.

small population of particles for few generations. We have further examined the behavior of the proposed method on larger search spaces. In a series of experiments the optimization method was employed to optimize individual camera parameters. More specifically, for an n camera setup, the parameter vectors had a dimension of $3n$.

On the dancer dataset the total number of parameters to optimize formed a vector of 24 dimensions (i.e., 8 cameras, 3 parameters per camera). The optimization procedure for this experiment, utilized 8 particles for 20 generations. The fitness and similarity-to-ground-truth curves found to be identical to the ones produced by the 10-frames experiment that was described in Section 6.3. A similar experiment was also conducted for the Kung-Fu girl dataset where 16 cameras were utilized resulting in a total parameter vector of 48 dimensions. For this experiment, the optimization algorithm required 200 generations of an 8-particle population to converge to results similar to the ones presented in Section 6.4.

6.8. Implementation and computational performance issues

The experiments were conducted on a PC with 6GB RAM, Intel 920 core i7 CPU and a Nvidia GTX 295 GPU. Confidence map computation and multiview silhouette estimates were implemented on GPU, using Nvidia's CUDA framework.[3] For the dancer dataset confidence maps were calculated at a rate of roughly 250 frames per second while on the Kung-Fu girl dataset we reached a rate of 800 frames per second. For foreground detection we employed the publicly available[4] CPU implementation of the method described in [29]. Foreground detection calculations for one generation of 8 particles and for the 10-frames experiment on the dancer dataset took 62 s. On the Kung-Fu girl dataset the corresponding time was 23 s.

[3] http://developer.nvidia.com/object/cuda_2_3_downloads.html.
[4] http://staff.science.uva.nl/zivkovic/Publications.

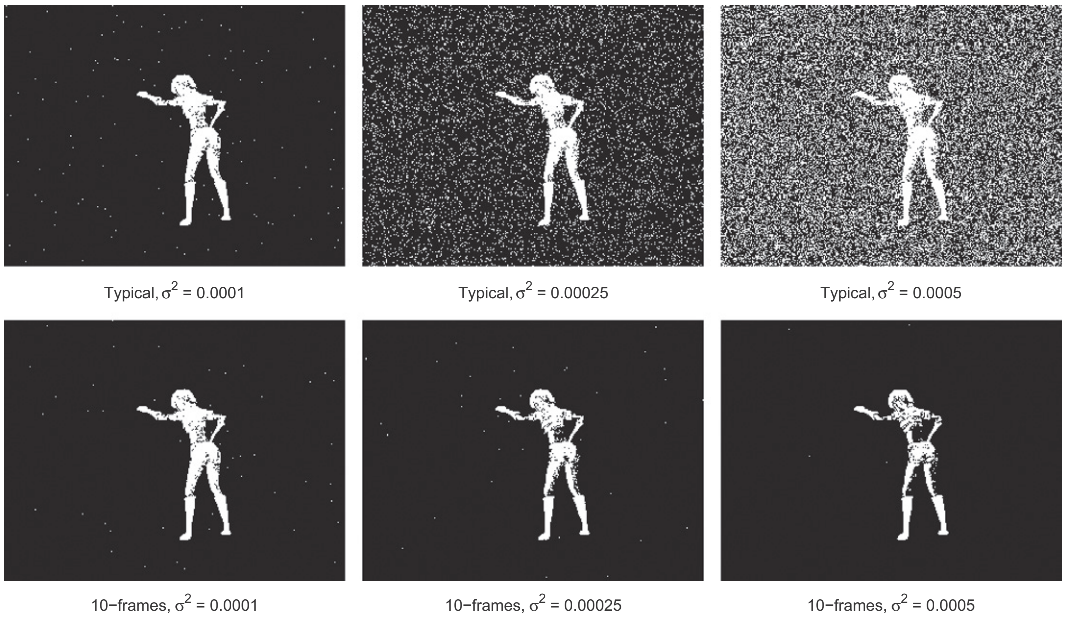

Fig. 11. The silhouettes returned by the typical and and 10-frames best foreground detection instances for the three noise levels of the conducted experiments. Results correspond to frame #138, view 8 of the Kung-Fu girl dataset.

Table 4
Parameter vectors estimated for the Kung-Fu dataset at various noise levels.

Parameter set	T_b	T_g	σ_0
Typical	16	9	11
$\sigma_1^2 = 0.00010$	16.8	19.5	11.0
$\sigma_2^2 = 0.00025$	33.5	23.7	13.0
$\sigma_3^2 = 0.00050$	34.8	30.2	28.0

7. Conclusions

We presented a novel algorithm for optimizing, in an unsupervised manner, the foreground detection parameters of a calibrated multicamera configuration. The proposed method successfully exploits information regarding the consensus of the setup on what constitutes foreground in an observed scene. By encoding this information in a fitness function, Particle Swarm Optimization optimizes the foreground detection parameters. Results showed a strong correlation of the fitness curve with the similarity-to-ground-truth curve, leading to the conclusion that the proposed definition of the fitness function is a good choice for the specific task. It was also shown that this method can be used, provided an efficient foreground detection implementation, for online applications.

The most important advantage of the proposed algorithm is that it does not require prior ground truth information or other kind of supervision. As such, it can be used as a tool for automatically adjusting the foreground detection parameters in frequently changing environments. The data used to evaluate our method have been captured in laboratory conditions. This has been motivated by the availability of ground-truth for these data sets, which is required for the quantitative evaluation of the proposed approach. It is expected that the benefits from the application of the proposed method in uncontrolled environments (i.e. outdoors surveillance) will be much greater due to the fact that, in such conditions, there is no single parameter set that performs well on average. Current and future work includes the extension of this approach to other interesting problems where multiview consensus can be exploited towards relaxing the requirement for ground truth data and/or supervision.

References

[1] P.J. Angeline, Evolutionary optimization versus particle swarm optimization: philosophy and performance differences, Evolutionary Programming VII, LNCS 1447 (1998) 601–610.
[2] T.M. Blackwell, P.J. Bentley, Don't push me collision-avoiding swarms, in: IEEE Conference on Evolutionary Computation, 2002, pp. 1691–1696.
[3] T. Bouwmans, F. El Baf, B. Vachon, Background modeling using mixture of gaussians for foreground detection – a survey, Recent Patents on Computer Science 1 (3) (2008) 219–237.
[4] K.M. Cheung, S. Baker, T. Kanade, Shape-from-silhouette of articulated objects and its use for human body kinematics estimation and motion capture, in: IEEE Conference on Computer Vision and Pattern Recognition, 2003.
[5] M. Clerc, The swarm and the queen: towards a deterministic and adaptive particle swarm optimization, in: Congress on Evolutionary Computation, vol. 3, 1999, pp. 1951–1957.
[6] M. Clerc, Discrete Particle Swarm Optimization. New Optimization Techniques in Engineering, Springer, Verlag, 2004.
[7] M. Clerc, J. Kennedy, The particle swarm-explosion, stability and convergence in a multidimensional complex space, IEEE Transactions on Evolutionary Computation 6 (1) (2002) 58–73.
[8] R.T. Collins, A.J. Lipton, H. Fujiyoshi, T. Kanade, Algorithms for cooperative multisensor surveillance, in: Proceedings of the IEEE, 2001.
[9] W. Du, J.B. Hayet, J. Piater, J. Verly, Collaborative multi-camera tracking of athletes in team sports, in: Computer Vision Based Analysis in Sport Environments (CVBASE), 2006, pp. 2–13.
[10] A. Elgammal, D. Harwod, L.S. Davis, Non-parametric model for background subtraction, in: International Conference on Computer Vision, FRAME-RATE Workshop, 1999.
[11] J.S. Franco, E. Boyer, Fusion of multiview silhouette cues using a space occupancy grid, in: International Conference on Computer Vision, vol. 2, 2005, pp. 1747–1753.
[12] B. Han, D. Comaniciu, L. Davis, Sequential kernel density approximation through mode propagation: applications to background modeling, in: Asian Conference on Computer Vision, 2004.
[13] J.H. Holland, Adaptation in Natural and Artificial Systems, The University of Michigan Press, 1975.
[14] J. Kennedy, R. Eberhart, Particle swarm optimization, in: IEEE International Conference on Neural Networks, 1995, pp. 1942–1948.
[15] A. Laurentini, The visual hull concept for silhouette-based image understanding, IEEE Transactions on Pattern Analysis and Machine Intelligence 16 (2) (1994) 150–162.
[16] W.N. Martin, J.K. Aggrawal, Volumetric descriptions of objects from multiple views, IEEE Transactions on Pattern Analysis and Machine Intelligence 5 (2) (1983) 150–158.
[17] T. Matsuyama, N. Ukita, Real-time multi-target tracking by a cooperative distributed vision system, in: Proceedings of the IEEE, 2002, pp. 1136–1150.
[18] T. Matsuyama, X. Wu, T. Takai, T. Wada, Real-time 3d shape reconstruction, dynamic 3d mesh deformation, and high fidelity visualization for 3d video, Computer Vision and Image Understanding 96 (3) (2004) 393–434.

[19] R. Mendes, J. Kennedy, J. Neves, The fully informed particle swarm: simpler, maybe better, IEEE Transactions on Evolutionary Computation 8 (3) (2004) 204–210.
[20] N.M. Oliver, B. Rosario, A.P. Pentland, A bayesian computer vision system for modeling human interactions, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (8) (2000) 831–843.
[21] M. Piccardi, Background subtraction techniques: a review, in: IEEE Conference on System, Man and Cybernetics, vol. 4, 2004, pp. 3099–3104.
[22] C.J. Van Rijsbergen, Information Retrieval, Butterworth Heinemann, Newton, MA, USA, 1979.
[23] A. Smolic, K. Mueller, P. Merkle, C. Fehn, P. Kauff, P. Eisert, T. Wiegand, 3d video and free viewpoint video-technologies, applications and mpeg standards, in: IEEE Conference on Multimedia and Expo, 2006, pp. 2161–2164.
[24] C. Stauffer, W. Grimson, Adaptive background mixture models for real-time tracking, in: IEEE Conference of Computer Vision and Pattern Recognition, vol. 2, 1999, pp. 246–252.
[25] P.N. Suganthan, Particle swarm optimiser with neighborhood operator, in: IEEE Conference on Evolutionary Computation, 1999, pp. 1958–1962.
[26] R. S Sutton, A.G. Barto, Reinforcement Learning: An Introduction, MIT Press, 1998.
[27] B. White, M. Shaw, Automatically tuning background subtraction parameters using particle swarm optimization, in: IEEE Conference on Multimedia and Expo, 2007, pp. 1826–1829.
[28] C. Wu, H. Aghajan, Collaborative gesture analysis in multi-camera networks, in: ACM SenSys Workshop on DSC, 2006.
[29] Z. Zivkovic, Improved adaptive gaussian mixture model for background subtraction, in: International Conference on Pattern Recognition, 2004.