Pattern Anal Applic (2011) 14:311–328
DOI 10.1007/s10044-011-0221-2

INDUSTRIAL AND COMMERCIAL APPLICATION

Detection of moving foreground objects in videos with strong camera motion

D. Szolgay · J. Benois-Pineau · R. Megret · Y. Gaestel · J.-F. Dartigues

Received: 4 August 2009 / Accepted: 11 May 2011 / Published online: 29 May 2011

Abstract In this paper, we propose a novel method for moving foreground object extraction in sequences taken by a wearable camera, with strong motion. We use camera motion compensated frame differencing, enhanced with a novel kernel-based estimation of the probability density function of background pixels. The probability density functions are used for filtering false foreground pixels on the motion compensated difference frame. The estimation is based on a limited number of measurements; therefore, we introduce a special, spatio-temporal sample point selection and an adaptive thresholding method to deal with this challenge. Foreground objects are built with the DBSCAN algorithm from detected foreground pixels.

D. Szolgay (✉)
LABRI, UMR 5800 CNRS, University of Bordeaux, Bordeaux, France
e-mail: szoda@digitus.itk.ppke.hu

D. Szolgay
Péter Pázmány Catholic University, Budapest, Hungary

J. Benois-Pineau
LABRI, UMR 5800 CNRS, University of Bordeaux, Bordeaux, France
e-mail: benois-p@labri.fr

R. Megret
IMS, UMR 5218 CNRS, University of Bordeaux, Bordeaux, France
e-mail: megret@enseirb.fr

Y. Gaestel · J.-F. Dartigues
ISPED, INSERM U 897, University of Bordeaux, Bordeaux, France
e-mail: yann.gaestel@isped.u-bordeaux2.fr

J.-F. Dartigues
e-mail: francois.dartigues@isped.u-bordeaux2.fr

Keywords Kernel-based density estimation · Motion detection · Wearable cameras

1 Introduction and motivation

Wearable video capture has been recently gaining popularity due to the availability of new low weight and low energy consuming hardware. From the pioneering works of Steve Mann [1] in the domain of wearable computing, who worked at concealing image acquisition and computing power inside non-invasive clothing, the technology has evolved to allow autonomous devices with a long battery life and image capture capabilities. One example is the SenseCam device [2], which can be hang around the neck due to its low weight, while recording images all day long. This type of device produces a new kind of video data, which brings new possibilities from the point of view of recording and using data acquired during everyday activities [3]. The advent of personal video capture using video cameras or mobile phones with cameras is one of the factors leading to a sharp increase of the quantity and ubiquity of such a data.

Automatic analysis approaches that were working on more traditional types of video, such as static, or motion-controlled cameras, now need to be adapted, in order to be able to automatically extract meaningful information from those new data. Segmenting foreground objects from the background is one such a basic module, which has a broad interest, as it is commonly used to bootstrap many higher-level analysis algorithms, such as object-of-interest detection and tracking. Strong motion and parallax, low quality of signal (reduced by motion blur) makes such videos very challenging. Moving object detection in these videos is not just a matter of camera motion compensation and then

detection of foreground pixels as if it was a still-camera video. As we show this in the paper, all steps in a "compensation-like" scheme in such a complex environment have to be studied very thoroughly. We will show formally that the proposed technique outperforms Stauffer and Grimson-like approaches on compensated frames.

We now present the application that motivated this work, before developing the generic foreground segmentation problematic it led to.

1.1 Motivation

This work is motivated by the development of new methods for the observation of patients suffering from dementia diseases (e.g. Alzheimer). The observation of patients during their daily activities helps diagnosing dementia stages and proposing targeted assistance. Such observations at home are not much developed now, because of the tremendous amount of time that it would require to be generalised. The stakes are high, as the recent PAQUID epidemiological study [4] has shown that the presence of some restrictions in Instrumental Activities of Daily Living (IADL) was correlated with the future appearance of a dementia related disease.

The IADLs are here considered to be the events of interest. They correspond to the interaction of the patient with objects during daily activities, such as preparing the meal and having dinner, washing dishes, receiving a phone call, opening doors, etc. Difficulties can arise at several cognitive levels, from the ability to control one's hands from a motor point of view to the elaboration and the correct realisation of strategies to accomplish the activities. Monitoring such difficulties requires acquiring enough pertinent information, which motivated the development of a wearable video capture device we first introduced in [5], and which is represented in Fig. 1. In this device, the camera is worn close to the patient's shoulder. Two types of camera can be used: a fish-eye camera and a standard button camera. Some examples of image snapshots acquired with such a device are shown in Fig. 2.

Wide-angle cameras proved to be more useful as they allow for better recording of events close to the camera such as IADLs, but the recorded video is difficult to

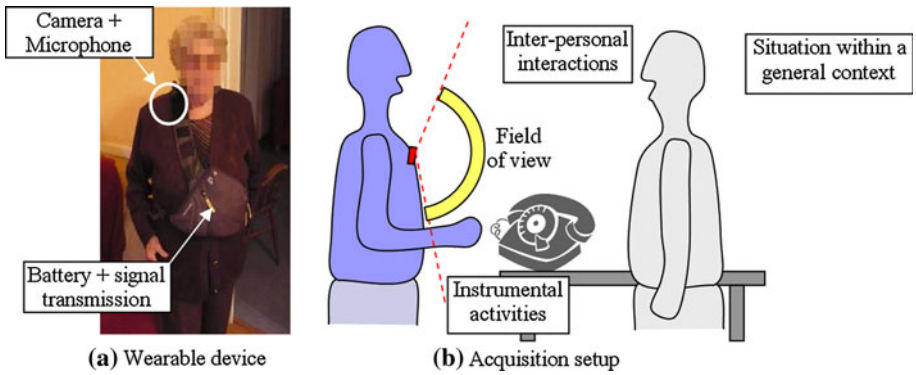

Fig. 1 Acquisition device and context

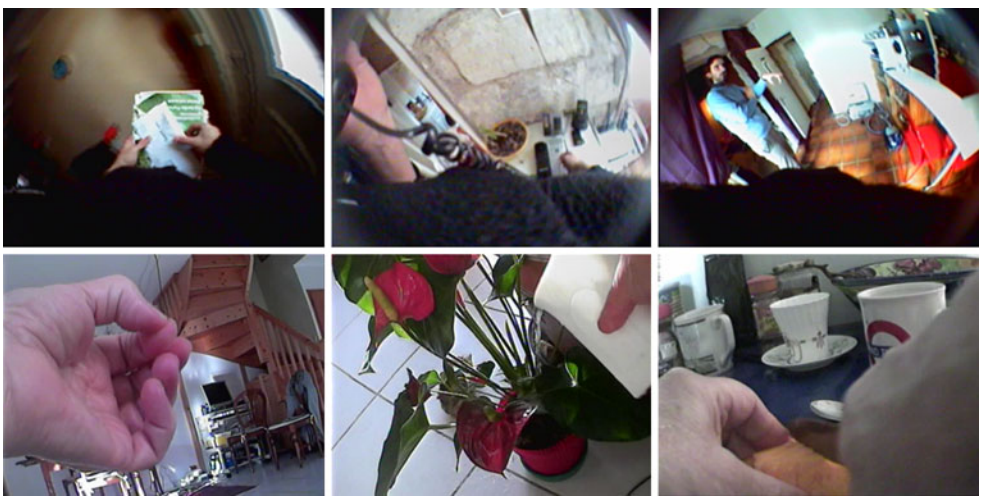

Fig. 2 Examples of image snapshots from acquired videos with fish-eye camera (*top line*) recordings, button camera recording (*bottom line*)

analyse as the image undergoes a strong non-linear deformation. The button cameras allow for a good understanding of the environment as well as the analysis of instrumental activities. Although at the present stage of our research we deal with button camera recordings, in the future we intend to use wide-angle cameras as well.

1.2 Problematics

The objective of the analysis of recorded videos (illustrated in Fig. 3) is to extract meaningful events related to the IADLs, in order to provide practitioners with video indexing assistance when using the videos for diagnosis purpose. We can identify two important low-level cues that are useful for IADL-related event analysis. The segmentation of the patient's hands and the detection of persons moving in front of the camera are strongly related to the instrumental activity or the situation of the patient, which helps understanding the context of an action. These two segmentations both benefit from a low-level segmentation that could separate the moving foreground (corresponding to hands or persons) from the background, which remains static in the 3D world.

One challenge of such a task resides in the relative instability of the camera position, which is strongly coupled to the movement of the wearer and the low quality of the frames due to motion blur.

In the rest of the paper, we tackle the problem of extracting foreground objects from a video taken by a moving wearable camera in the conditions of strong and unpredictable camera motion.

To handle these difficult conditions we propose motion compensated frame differencing and a kernel-based background model estimation, with a specific spatio-temporal selection of measurements to handle the sparseness of the data.

Hence, in Sec. 2, the problem of foreground object detection is discussed with the current state-of-the-art solutions and the general scheme of the proposed method is presented. In Secs. 3, 4 and 5 the details of the proposed method are described. Section 6 summarises the results and Sec. 7 draws the conclusion and shows perspectives for future work.

2 Foreground object detection

The detection of moving objects is an important task for video surveillance and computer vision systems. After reviewing relevant works amongst the numerous research papers on the subject, we will present an overview of the proposed approach.

2.1 Related work

Segmentation of the foreground moving objects is an intensively researched topic [6–19]. In most of the cases, there is no available information about the foreground so it cannot be modelled directly. Instead, the background is modelled by using the information of consecutive frames from the past, where the background considered to be unchanged.

The modelling of the background even in the case of still cameras raises a lot of difficulties like occlusions, shadows cast by the foreground objects, change of illumination, moving background elements (trees, flags, waves). The objects' silhouettes, obtained by background subtraction, are often not accurate, which affects the performance of higher-level applications.

In the well-known work of Stauffer and Grimson [6], the authors present a method that can deal with some of the above-mentioned problems like lighting changes, repetitive motion, and long-term scene changes. They use an adaptive background model for motion segmentation. The colour distribution of each pixel is estimated with a Gaussians Mixture Model (GMM). The Gaussian distributions are then evaluated to determine which are most likely to be the model of the background. For improving this idea further, recursive equations are used in [7, 32] and [8] to constantly update the parameters and to simultaneously select the appropriate number of components in the GMM for each pixel.

A similar approach using maximum likelihood decision rule together with GMM for background model was proposed in [9]. Its advantage is smoother object detection, due to a regularisation based on Markov Random Field modelling.

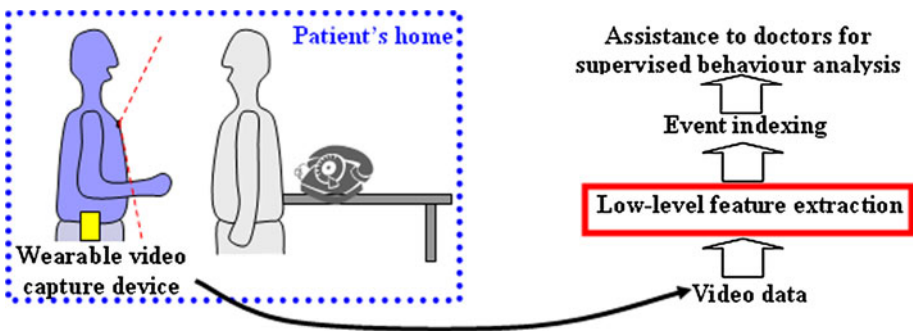

Fig. 3 Principle of multi-level analysis for the video acquired using the wearable camera

In [10] an approach is proposed to consistently label people and to detect human–object interactions using mono-camera surveillance video. After background subtraction [11], the non-stationary objects are used to build a robust appearance-based correlogram model combined with histogram information for each human and object in the scene. This method is capable of detecting when people merge into groups and to segment them even in case of partial occlusions. It can also detect when a person deposits or removes an object. People who have left the scene and reappeared later can be identified based on their stored colour model.

The periodic changes of the background represent one of the main difficulties in case of surveillance cameras. The traditional background models fail to recognise the changing background parts and treat them as foreground. The authors of [12] propose a method that can deal with periodically changing background elements, by modelling the dynamic characteristics using the optical flow parameters in a higher dimensional space as a feature. The background model is calculated by kernel density estimation with data-dependent bandwidth.

All these research works deal with a stationary camera, when the background on 2D sequenced scene is static and can be efficiently modelled. The extraction of moving objects in the case of a moving camera is even more challenging. There are two different motions in an observed scene: the ego-motion of the camera and the motion of the object. To extract the object's motion, the camera motion has to be estimated and compensated [13].

In [14] the authors present a surveillance system with a moving camera. Its motion is estimated with feature tracking. The moving object detection is done via background compensation. Here the camera is an outdoor surveillance camera, with more planar view and less abrupt camera motion, than in our "wearable" case.

In [15] a system is introduced which uses a single camera to extract human motion in an outdoor environment. The camera is installed on a mobile robot and the motion of the camera is compensated using corresponding feature sets and outlier detection. The positions of moving objects are estimated using an adaptive particle filter and Expectation–Maximisation (EM) algorithm.

A similar system is described in [16], where the authors propose an integrated computer vision system designed to track multiple humans and extract their silhouette with a pan-tilt stereo camera. The detection of foreground objects is performed by camera motion compensation and disparity segmentation.

The authors of [17] propose a framework, derived from a perceptual grouping principle, namely the Helmholtz principle. This principle basically states that perceptually relevant events are perceived because they deviate from a model of complete randomness. Detection is then said to be performed a contrario: moving regions appear as low probability events in a model corresponding to the absence of moving object in the scene. However, in the presence of strong parallax, some parts of the static background may be considered as a moving object.

The method described in [18] deals with detection of motion regions in video sequences observed by a moving camera, in the presence of strong parallax, due to 3D static objects. The proposed method classifies each image pixel into planar background, parallax or moving regions using 2D planar homographies, epipolar constraint and a so-called structure consistency constraint. The method was tested on different outdoor sequences with encouraging results, but a known limitation of the algorithm is that, it cannot handle abrupt camera motion, which is necessary in our case.

In [19] the background is modelled by one single probability density function using a nonparametric density estimation method over a joint domain-range representation of image pixels. The foreground is also modelled based on previous detections and used competitively with the background model. The strength of the method is its capability to handle dynamic textures, cyclic motions and "nominal" camera motion. They use fixed cameras, where the camera motion comes from the effect of wind or trembling of the ground. The magnitude of these motions can be very strong but the scene does not change and the number of measurements is not limited as strictly as it is in our case.

As we can see from this, analysis deal with static cameras. In the case of a strongly moving camera, the problem remains complex. With regard to the state-of-the-art, our contribution consists in intelligent combination of fundamental methodologies for the detection of moving foreground objects in wearable video settings. Hence, to compensate motion blur frames, we used hierarchical block matching, which is more robust to blur than conventional pixel-based methods.

Then to filter remaining motion compensation errors we built a so-called "Modified Error Image". The latter allows of application of costly kernel-based estimators to a reduced number of pixels in image plane.

In estimating the density we proposed a smoothed bandwidth estimation on spatio-temporal patches, thus taken into account both temporal history and spatial context.

In decision-making process, we use an efficient heuristics for probability approximation. Finally, we apply a known clustering method DBSCAN in a mixed motion and colour space to extract foreground objects.

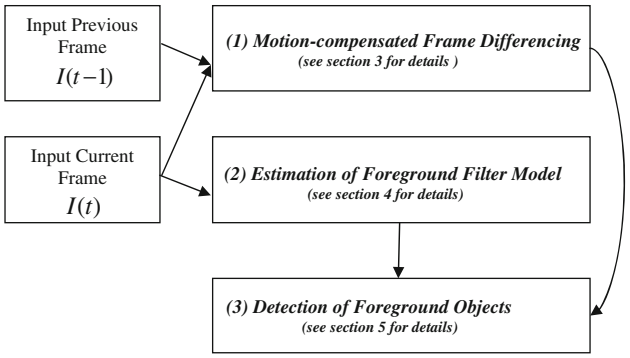

Fig. 4 Diagram of the foreground object extraction method

2.2 General scheme of moving foreground object detection

The method we propose consists of three steps: (1) Motion compensated frame differencing, (2) Estimation of foreground filter model, (3) Detection of moving objects (see Fig. 4).

The compensation of camera motion is a must in step (1). After the compensation, the two frames have the same coordinate system and an error image can be calculated as a difference of compensated frames. This error image should contain only the foreground regions.

Due to changes in the perspective, quantization error and other sources of noise (transmission noise, strong motion blur, etc.), the foreground contains a lot of false positives. To eliminate this noise we use a foreground filter model (step (2)), built on a Modified Error Image (MEI) resulting from step (1). Because of the movement of the person who is wearing the camera and changes of capturing conditions in a natural environment, this model has to be continuously updated. The last step is the Detection of the moving objects. Based on the model from step (2), the background pixels are eliminated from the MEI and then a density-based clustering (DBSCAN) is applied to the remaining foreground pixels to build foreground objects.

3 Motion-compensated frame differencing

In our case of a non-static background, to know what parts of the picture are changing because of the camera motion and what parts are changing independently, the camera motion has to be estimated.

For correctly aligning video frames, a Hierarchical Block-Matching (HBM) algorithm [20, 21], was used. It allows estimating strong motion and has proven to be the best motion estimation approach in video coding applications, and it is robust to local motion blur. The principle of HBM consists of dividing the current video frame $I(t)$ into a set of blocks. Then for each block its best match is searched in frame I_{ref} by minimising a sum of absolute difference criterion, which is a function of a frame difference: $\Delta I(t) = |I(t) - I(t-1)|$. The difference of block centre positions $\vec{d} = ((x_t - x_{t-1}), (y_t - y_{t-1}))$ is called a displacement vector. We refer the reader to [20, 21] for details of HBM, which allows estimating large displacements, up to 30 pixels in our case (see Fig. 5).

The resulting motion vectors are then used as initial measures for a robust motion estimator [22] allowing for the rejection of outliers—and obtaining the global camera motion model:

$$\vec{d}(c_x, c_y) = \begin{pmatrix} a_1 \\ a_4 \end{pmatrix} + \begin{pmatrix} a_2 & a_3 \\ a_5 & a_6 \end{pmatrix} \begin{pmatrix} c_x \\ c_y \end{pmatrix}, \quad (1)$$

where $\vec{d}(c_x, c_y)$ is the displacement vector of the pixel block of centre (c_x,c_y).

3.1 Creation of the modified error image

After camera motion compensation our goal is to separate the moving objects from the background containing noise. Here we resort to the family of methods, which model the probability density function of the background pixels and use it in the decision-making. The approach proposed will neither use the original frame entirely, nor a simple frame differencing. We propose a new measurement scheme building a signal we call "Modified Error Image" (MEI).

After estimating and compensating the camera motion, the two consecutive images are aligned in the same coordinate system so that a frame difference can be calculated.

Let $I(t-1)$, $I(t)$ be two consecutive frames. With (1) we transform $I(t-1)$ according to the camera motion between $I(t-1)$ and $I(t)$. We use this motion-compensated image, $\tilde{I}.(t-1) = I((t-1), (x + dx, y + dy))$ to calculate an error image $E(t)$, which shows the pixels moving independently from the camera:

Fig. 5 Three consecutive frames from a wearable outdoor video with strong motion

Fig. 6 *Top left*: Previous frame ($I(t-1)$), *Top right*: Current frame (I(t)), *Bottom left*: Difference frame, *Bottom right*: Difference with motion compensation (E(t))

$$E(t) = \left|\tilde{I}(t-1) - I(t)\right| \tag{2}$$

Figure 6 shows how the motion compensation enhances the result of frame differencing.

In the case of ideal camera motion compensation and in the absence of noise, the pixels with non-zero motion magnitude would be those that have ego-motion. In reality, due to changes in the perspective, quantization error and motion blur, the highly contrasted contours in the scene will never be perfectly compensated. Thus, the resulting error image will contain not only the pixels of a moving object, but false positive pixels too, making the direct detection of moving objects' pixels impossible.

Hence, we propose to create a new, modified error image, on which the differentiation between "static" artefacts and moving objects could be done along the time.

The differentiation of static pixels and moving ones based only on grey level values is limited and we do not have any a priori information on the objects, hence for better discrimination, we would like to fully exploit the available colour information. We propose to use the colour information of the original image, thus the MEI at time t, will contain the colour information of the original frame on those (x,y) pixels, where, the value of the error image, $E(x,y,t)$ is significant. More formally, the modified error image, E^{m} is built as follows:

$$E^{\mathrm{m}}(x,y,t) = \begin{cases} I(x,y,t) & \text{if} \quad E(x,y,t) > th_E \\ 0 & \text{else} \end{cases} \tag{3}$$

where I is a 3 channel frame taken by the camera at time t, E is the grey scale motion compensation error at time t. The threshold th_E helps filtering irrelevant, but non-zero values from the error image. We experimentally fixed it to $th_E = 10$. Hence, the modified error image contains colour information of the original frame, which will be used at the decision-making step. Figure 7 shows an example of the MEI.

Contrary to approaches, which use the whole original colour frames for moving foreground detection, the MEI drastically saves computational workload: only pixels of original frames, where motion compensation error is strong, will be considered. According to our experiments, in average only 17% of the pixels of the original frame have to be processed. Furthermore, with motion compensated frame differencing and MEI building, we properly eliminate complex background motions, thus reducing the overall complexity of object detection. While in [12] for each pixel a Probability Density Function (PDF) based decision is made on its label foreground/background, in our approach the PDF based decision is done for false foreground removal only, as it is explained in the following section.

4 Estimation of foreground filter model

The objective in this phase is to estimate a PDF of the colour distribution associated to the background for each separate pixel on the MEI. This estimation is based on a short-term image history, in order to consolidate observations over several frames, after motion compensation. The assumption is that a pixel corresponding to a

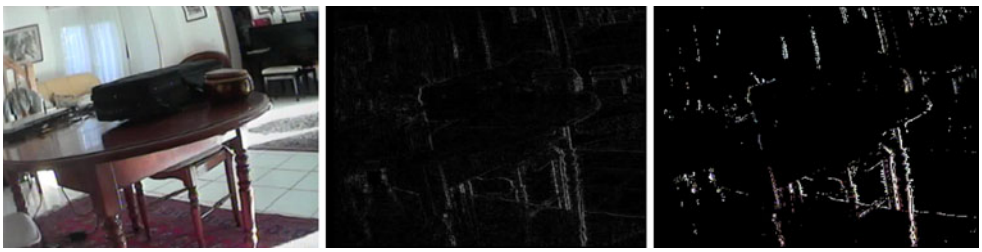

Fig. 7 The original image (*on the left*) taken by the camera, grey scale error image (*in the middle*), and the MEI (*on the right*)

moving foreground object will have varying colours over such a time interval, whereas a pixel belonging to the background will have more stationary colours. Comparing the current pixel colour to its corresponding background model therefore allows refining detection by taking into account more frames. In order to decrease the computational complexity, this estimation is done only for pixels that have been segmented as potential foreground pixels during the motion-compensated frame-differencing phase. We assume that large homogenous foreground areas (e.g. car, bus) do not appear in a home environment and we do not have to face the so-called foreground aperture problem. The rest of this section is devoted to present how to estimate such a meaningful PDF.

4.1 Measurement matrix

In order to estimate the PDF for background pixel values, we build a measurement matrix M. This matrix contains the information of the original frames, in n consecutive time instances and it is continuously updated along the time. Because of the unpredictable camera motion, a short temporal window is used for gathering frames to M (For the presented experiments a 15 frame long time window was used). In this way, we can ensure that the frames in M have large overlapping parts, and are less affected by motion compensation errors.

Updating at time t means adding the information of the current frame at time t to the measurement matrix of the previous time instance, $t-1$.

$$\begin{aligned} M(x,y,t) &= \Theta_t M(x,y,t-1) \cup I(x,y,t) \\ M(x,y,1) &= I(x,y,1) \end{aligned} \tag{4}$$

where the operator $\cup$ means adding new frame of measurements, while the oldest frame is being removed. Thus the number of frames in the matrix remains always the same. The operator Θ_t stands for the affine transformation with the estimated parameters of camera motion between time instance $t-1$ and t, (1). Applying this transformation, we compensate all frames in the matrix to the reference frame, the current one.

4.2 Estimation of background colour model

The measurements, stored in M, are used to estimate the probability that an incoming pixel belongs to the background. We use a kernel-based density estimation method [23].

4.2.1 Kernel-based density estimation

Density estimation is the construction of an estimate of an unobservable probability density function, based on observed data. The data are usually thought of as a random sample, drawn from an unobservable density function. Perhaps the most popular approaches for density estimation are Kernel-based Density Estimation (KDE) and Gaussian Mixture Model (GMM). We have chosen KDE over GMM since it is more reliable in case of low number of data available, as stated in [24]. Namely, we use K_n nearest neighbour approach (see [25], p. 174 for the detailed description of the method).

The aim of this estimation method is to extrapolate the measured data into a regular density function. For the extrapolation, kernel functions, placed at each measurement point, are used with a smoothing parameter. We have considered using marginal or joint probability density estimation (see Sec. 6 for test results).

Let $v_1, v_2, \ldots, v_n$ be a set of d-dimensional, i.i.d. sample points in $\mathbf{R}^d$, drawn from a random variable that follows a probability density function f. Let K: $\mathbf{R}^d \rightarrow \mathbf{R}$ be a kernel satisfying the following conditions:

$$\int_{R^d} K(v)dv = 1, \tag{5}$$

$$\int_{R^d} vK(v)dv = 0, \tag{6}$$

$$\int_{R^d} vv^T K(v)dv = 1 \tag{7}$$

and $K(v) \geq 0$. The (5, 6, 7) together with non-negativity define K as a zero-mean, unit-variance PDF. In our case $d = 3$ as v is a colour vector of a pixel.

We can define the kernel-based approximation of function f at the estimation point v for a given n as follows:

$$\tilde{f}(v) = \frac{1}{n \cdot \|H\|^{1/2}} \sum_{i=1}^{n} K(H^{-1}(v - v_i)) \tag{8}$$

where n is the number of v_i sample points, and H is a smoothing parameter (bandwidth matrix), which determines the width and the height of the kernel function. For the sake of simplicity, we use a diagonal bandwidth matrix:

$$H = \begin{bmatrix} \sigma^2[\mathrm{c1}] & 0 & 0 \\ 0 & \sigma^2[\mathrm{c2}] & 0 \\ 0 & 0 & \sigma^2[\mathrm{c3}] \end{bmatrix} \tag{9}$$

where each σ^2 represents the bandwidth for a colour channel.

The selection of the kernel function and the bandwidth parameter are obviously very important, since they both have a strong influence on the accuracy and the smoothness of the PDF estimate.

4.2.2 Selection of bandwidth and kernel function

The kernel bandwidth can be either fixed or varying. Choosing a fixed bandwidth is simple and computationally efficient, but usually less accurate than variable bandwidth, especially when there are only a few sample points available.

To adapt the bandwidth to the sample data, we considered two traditional ways for variable bandwidth selection: the balloon estimator and the sample-point estimator [23]. In the case of the balloon estimator, the bandwidth is a function $H = H(v)$ of the estimation point. Usually it depends on the sample-point neighbourhood of the estimation point.

The sample-point estimator means that the kernel's bandwidth is a function $H = H(v_i)$ of the sample point v_i on which it is centred, thus all the kernels building up the density might have different bandwidth.

In this work, we use a sample-point density estimator, which can be defined as follows:

$$\begin{aligned} f_s(v) &= \frac{1}{n} \sum_{i=1}^{n} K_{H(v_i)}(v - v_i) \\ &= \frac{1}{n} \sum_{i=1}^{n} \frac{1}{\|H(v_i)\|^{1/2}} K(H^{-1}(v_i)(v - v_i)) \end{aligned} \tag{10}$$

where n is the number of measurements, and $H(v_i)$ is the sample-point bandwidth matrix associated to the ith sample point v_i.

A common choice for the bandwidth calculation consists of using the distance of the sample point v_i from its kth nearest neighbour. However, in our case, the number of sample points is low and this kind of calculation might give false result, as it was pointed out in [26].

Choosing a high value for K would mean that the distance from the kth sample point to the point v_i can be high even if there are (at maximum $k - 1$) other sample points near to v_i. In this case the information that there are other points near is completely lost. Choosing a low value for k, would mean that the distance might be low even if there are not many other samples near to the point, which makes the calculation sensitive to noise. Choosing a right k in case of few measurements is not always possible.

To handle this problem we use the distance from all the k nearest neighbours, instead of using the distance from the kth alone. The $\sigma_i[c]$ parameter is calculated as a variance of the k nearest neighbours around the measurement v_i:

$$\sigma_i^2[\mathrm{c}] = \frac{1}{k} \sum_{j=1}^{k} \left(v_i[c] - v_j[c]\right)^2 \tag{11}$$

where v_j is the jth nearest value to v_i in the measurement matrix c is the index of colour channel. For the estimate $\tilde{f}(v)$ to converge to the true unknown PDF, the following should be satisfied: $k(n)/n \to 0$ when $n \to \infty$. We use $k = \sqrt{n}$, where n is the number of available measurements.

It is generally accepted that the choice of the bandwidth is more important than the choice of the kernel function [27], although when the number of sample points is limited, the kernel function might have higher influence on the estimation. Several kernel functions (e.g. Epanechnikov, Gaussian, uniform, triangle, quadratic, etc.) were tested and the Gaussian function proved to be the most suitable for our task (see Sec. 6.3.4).

Choosing the Gaussian as kernel function the density estimator will be:

$$f_{x,y}(v) = \frac{1}{n(2\pi)^{d/2}} \sum_{i=1}^{n} \frac{1}{\|H(v_i)\|^{1/2}} e^{-\frac{1}{2}\left((v-v_i)^T H^{-1}(v_i)(v-v_i)\right)} \tag{12}$$

4.3 Spatio-temporal selection of the measurement points

To cope with the lack of data, due to limited temporal history, we propose a new concept of a spatio-temporal PDF, which will be explained in this section. We call it spatio-temporal according to the choice of sample points: we use both spatial neighbourhood and temporal history.

In [6] and [12] the sample points at given (x,y) coordinates are the n previous measurements taken at the same (x,y) position: $(v(x,y,1), v(x,y,2),\ldots,v(x,y,n))$. However, when the camera is moving the case is different: even after motion compensation the real background scene position that corresponds to the (x,y) pixel in one frame, might move a little, due to minor errors of camera motion compensation or quantization. Assuming that this error is random, the use of a small (x,y) centred patch can solve the problem.

We have tested two different methods for gathering measurements from the patch. The straightforward idea is to use all points from it. In this case, some noise might be added to the data but we can increase the number of measurements significantly.

The other idea is to use that pixel from the patch which might correspond to the pixel in question. To find it we look for the closest pixel in colour space. This way the data contains less noise but the number of measurements remains low.

We have made experiments with both methods and based on the results we have chosen the first approach (see the "Results" section for more detailed explanation).

Based on the values of the measurement matrix, probability density functions (PDF) are built for each non-zero (x,y) pixel of the current modified error image using (12), where v_i is the previously measured value of the (x,y) point, obtained from the M matrix through spatio-temporal selection, n is the number of measurements, $H_i = H(v_i)$ is the bandwidth matrix (see (9) and (11)).

We have to note that n is an effective maximal number of the non-zero measures available for PDF building. In practice, the number of available measurements can change for each pixel. In case when there is no measurement for a pixel we do not apply the kernel-based filter, thus these pixels will remain on the Filtered Error Image as foreground.

5 Detection of foreground objects

Once the PDF has been built for each pixel in the current MEI, we can proceed to the detection of moving foreground objects. Here the pixels will be first classified as belonging to the foreground or background on the basis of the PDFs characteristics. Then the detected pixels will be grouped into clusters (moving objects) on the basis of their motion, colour and spatial coordinates in the image plane.

5.1 Classification of foreground/background pixels

The $f_{x,y}(v)$ function is a PDF that shows how likely the pixel (x,y) takes a value v. Based on this likelihood we want to divide the domain R of all possible v values into two parts: R_1 and R_2. R_1 is associated to the background colours and R_2 to the foreground colours. If we measure a value which is in R_1, it will be classified as background; otherwise it will be classified as foreground. The union of R_1 and R_2 has to be equal to R which is the whole domain.

When classifying a measured value into background or foreground two kinds of mistakes can be made: classify a background point as foreground and classify a foreground point as background. Let p_1 be the background PDF and p_2 be the foreground PDF. Then the probabilities of misclassification are:

$$P(2|1,R) = \int_{R_2} p_1(v)dv \tag{13}$$

$$P(1|2,R) = \int_{R_1} p_2(v)dv \tag{14}$$

In our case $P(2|1,R)$ is the probability of false detection of an object pixel and $P(1|2,R)$ is a missed detection of an object pixel. If both PDFs would be known, we could find an optimal division of R that minimises the two kinds of error: $P(2|1,R) + P(1|2,R)$. However the PDF of the foreground is not known in our case, hence we propose a threshold-based decision scheme using only the background PDF.

5.1.1 Adaptive threshold calculation

Our goal now is to keep $P(2|1,R)$ small, while ensuring that the territory of the background remains as small as possible. We define R_1 and R_2 as follows:

$$R_1 = \left\{ v \in R \middle| f_{x,y}(v) \geq T_{x,y} \right\}, \tag{15}$$

$$R_2 = \left\{ v \in R \middle| f_{x,y}(v) < T_{x,y} \right\}, \tag{16}$$

where $f_{x,y}(v)$ is the PDF of the background and $T_{x,y}$ is a threshold. The higher the $T_{x,y}$, the smaller the R_1, and the bigger the $P(2|1,R)$. By calculating the integral of $f_{x,y}(v)$ over R_2 $P(2|1,R)$ can be controlled and we kept it under a predefined α (misclassification probability of the background):

$$\int_{R_2} f(v)dv = P(2|1,R) < \alpha \tag{17}$$

Determining T based on this integral requires too much computational power. To avoid this, we use a simple heuristic. First let us introduce a heuristic property, the Efficiency Measure (EM) for a PDF, which shows how efficiently the PDF can be covered by a closed sub-domain:

$$\mathrm{EM} = \frac{\int_{|r|} f(x)dx}{|r|} \tag{18}$$

where $|r|$ is the measure of the sub-domain. Our heuristic claims that the average height of the function in the sample points v_i (where the Gaussians are centred) is proportional to EM (and thus inversely proportional to the territory needed to cover it).

$$\frac{\sum_{i=1}^{n} f(x_i)}{n} \propto \mathrm{EM} \tag{19}$$

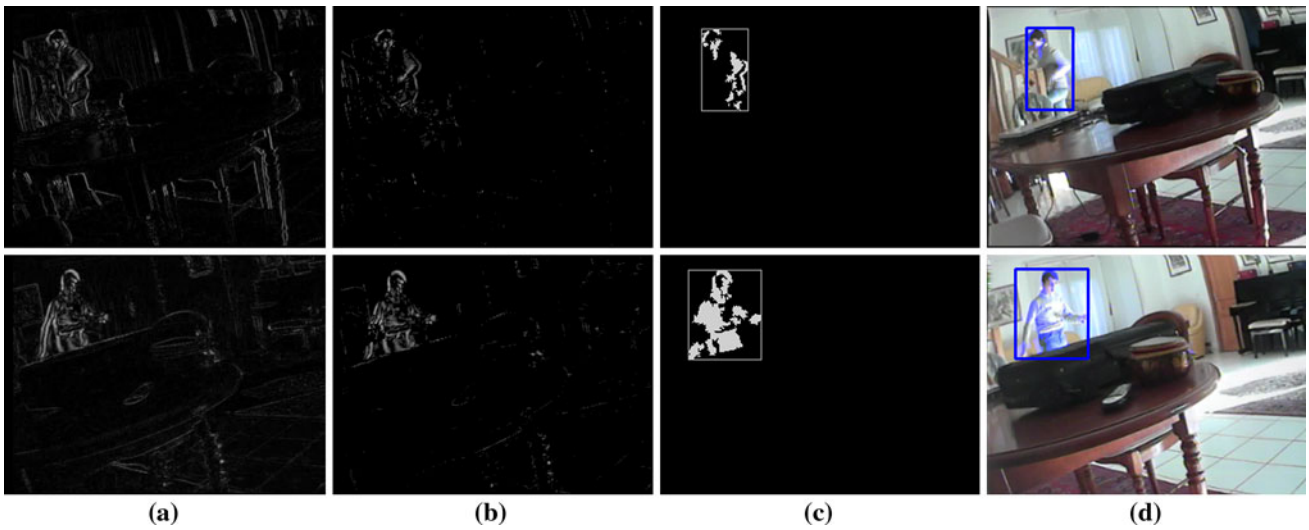

Fig. 8 The main steps: error image (**a**), filtered error image (**b**), foreground objects detected by DBSCAN (**c**) and the foreground objects on the original image (**d**)

If we choose the threshold to be proportional to that average, than it will be higher (and R_1 will be smaller) if the EM is high, and lower (R_1 will be bigger) when the EM is small. It means that the threshold will be adapted to the shape of the PDF. After this consideration we have chosen the $T_{x,y}$ threshold as follows:

$$T_{x,y} = \lambda \cdot \frac{\sum_{i=1}^{n} f_{x,y}(v_i)}{n} \tag{20}$$

where λ is a constant, n is the number of the available measurements and $v_i \in M(x, y)$.

5.1.2 Decision-making rule

Once the adaptive threshold has been defined, the classification of pixels into foreground/background is straightforward. We perform it on the modified error image $E^{\mathrm{m}}(x,y,t)$: if the new value v has a high probability at the given (x,y) position then we consider it as part of the background and eliminate it from the error image:

$$E_f^{\mathrm{m}}(x, y, t) = \begin{cases} E^{\mathrm{m}}(x, y, t) & \text{if} \quad f_{x,y}(v) \leq T \\ 0 & \text{otherwise} \end{cases} \tag{21}$$

The resulting $E_f^{\mathrm{m}}(x, y, t)$ is the filtered error image, which contains the points of the foreground.

The typical results of this classification are depicted in the left-most column of Fig. 8. Due to the noise and errors in motion estimation, this detection result still contains some noise, such as isolated pixels. Furthermore, the foreground pixels are spread over the frame while in the context of our problem, the moving objects should represent compact areas in the image plane.

We can reasonably suppose that with the high frame rate we have, the pixel displacements of objects are similar. Hence we propose to cluster detected foreground points in a mixed feature space, supposing that an object will be represented by one single cluster or by a set of clusters close to each other in the image plane.

5.2 Clustering of the foreground points with DBSCAN

To find moving objects' silhouettes and eliminate the remaining noise, we used a clustering algorithm, called DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [28] in a mixed feature space R^l. In this space with $l = 7$ dimensions, each foreground point is described with a feature vector $X = (x, y, C_1, C_2, C_3, dx, dy)^T$ which contains the x,y coordinates, the colour coordinates C_1, C_2, C_3 in normalised RGB space and the coordinates of a displacement vector dx, dy expressing pixel motion. Intuitively this means that the points that are close to each other, moving together and have similar colour will be put in the same cluster.

DBSCAN is a density-based clustering algorithm that can separate arbitrary shaped clusters. The main advantages of DBSCAN are: it does not require knowing a priori the number of clusters, does not have a bias towards a particular cluster shape or size and it is resistant to noise [28, 29]. On the other hand, it does not work well on high dimensional data or a dataset with varying density. For all these reasons—presence of detection noise, low dimensionality of the feature space, arbitrary shape of presumed clusters we found it to be convenient for our problem.

The results of the clustering can be seen in Fig. 8. Column c, contains the clusters obtained with DBSCAN

from a raw foreground detection results (column b). One can see that a lot of detection has been filtered. The bounding boxes of the clusters are superimposed on the original frame in the right-most column.

6 Results

The presented results were obtained on healthy volunteers and real patients. In order to keep the conditions of the experiment ergonomic for the observed subjects, the videos show their standard everyday conditions. They usually stay alone at home. This is why the sequences containing moving objects (persons, animals) are very rare. In the corpus of duration of 9 h 17 min moving objects (persons in our case) only occur occasionally for short periods of a few seconds. Hence to construct the ground truth for the tests of our method, we mainly used these short sequences. Some key frames and the corresponding results are presented in Fig. 9. A sample of the dataset and the belonging ground truth on healthy volunteers is publicly available at: http://www.labri.fr/projet/AIV/projets/peps/index.php?id=50.

In order to assess the false detection rate of our method experiments on sequences without moving objects will also be presented.

6.1 Evaluation metrics

To evaluate the performance of the method we used F-score [30]:

$$F = \frac{2}{\frac{1}{\mathrm{Re}} + \frac{1}{\mathrm{Pr}}} \tag{22}$$

where *Re* is the detection rate (recall) and *Pr* is the positive rate (precision).

The *Recall* and *Precision* were measured using two kinds of Ground Truth (GT) data. For comparison with a base-line method [7] we used handmade rectangular shaped GT. Every pixel inside the GT area was considered as foreground and every pixel outside the GT rectangular as background.

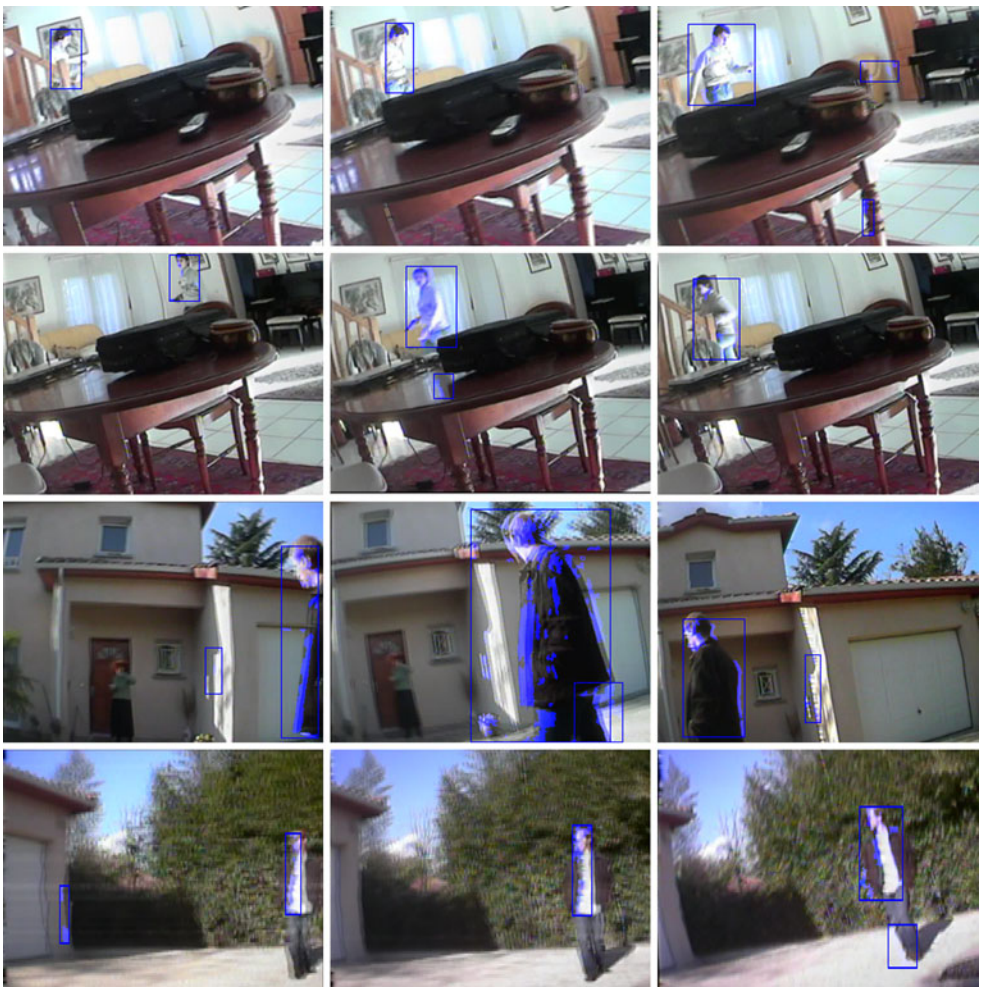

Fig. 9 Example of pictures from the tested sequences (From *top* to *bottom*: Francois 1, Francois 2, Daniel 1, Daniel 2) with detection results

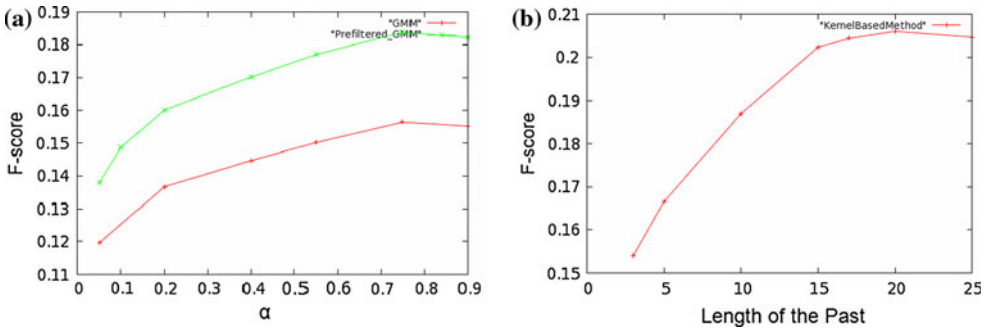

Fig. 10 Results of the GMM alone and pre-filtered as a function of learning parameter (α), and results of Kernel-based method as a function of the number of previous frames considered for building the PDF

During the search for the best parameters we used a modification of the above described GT. The modifications will be explained in details later.

6.2 Comparison with a base-line method: Gaussian mixture model

As a base-line method we used a variety of Stauffer and Grimson's GMM method [8]. This alternative method is based on [7], with additional selection of the number of the Gaussian components: [31]. Both GMM and Kernel-based model were tested on motion compensated images. Here GMM is used as a background model: the pixels that do not fit to the model will be foreground pixels. The maximum number of Gaussians in the method [8] was fixed as $K = 4$. The initial bandwidth for a new mode was chosen $\sigma = 11$ and the complexity reduction prior constant was CT = 0.2. The method was used without shadow detection. The results of the detection as a function of the learning parameter α, which tunes the update of Gaussians, are given in Fig. 10a. For detailed description of the method see [7, 8]!

The best result obtained for GMM in terms of F score was at $\alpha = 0.75$ where $F = 0.156$.

In order to test the effectiveness of the Modified Error Image (3) as pre-filter, we have taken the MEI as initial foreground mask and confirmed it by detected pixels with GMM method applied to motion compensated frames. This is the same concept as the Kernel-based density estimation was used in our method (see Sec. 5.1.2). The received F score was higher with this pre-filtering concept: $F = 0.183$. Hence the MEI not only saves a lot of computational time, but increases the effectiveness of the foreground detection.

Table 1 Peak F scores for the base-line and the Kernel-based method

	GMM	GMM filter	Kernel-based filter
Peak F score	0.156	0.183	0.206
Precision	0.0938	0.114	0.152
Recall	0.469	0.458	0.316

Exchanging the GMM with the proposed Kernel-based estimation with Gaussian kernel shows further improvement: $F = 0.206$.

Figure 10 shows the results of GMM, GMM filter and Kernel-based filter methods as a function of dependency on the previous frames. The peak F scores are summarised in Table 1. The best result in case of Kernel-based filter was obtained at $n = 20$. Since the F score does not change much between $n = 15$ and 20, to save computational power we decided to use $n = 15$. The F score at $n = 15$ is 0.202.

6.3 Step by step validation of the kernel-based filtering method

As Table 1 shows, Kernel-based method gives better results in the same circumstances; hence, it was chosen over GMM for calculating PDF for each candidate foreground point, based on previous measurements. The question is what parameter set (colour space, measurement point selection, patch size, etc.) is the most suitable for our task, where a special difficulty is presented by the strongly limited number of measurements.

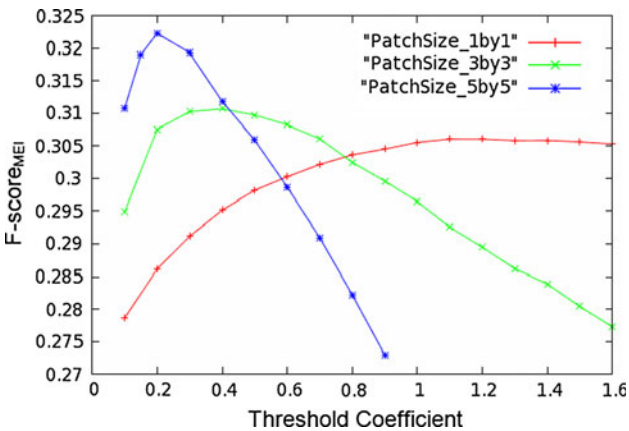

Fig. 11 Results obtained with different patch sizes: 1 × 1, 3 × 3, 5 × 5

Table 2 Peak F $\text{scores}_{\text{MEI}}$ obtained with different patch sizes

Patch size	1 × 1	3 × 3	5 × 5
Peak F $\text{score}_{\text{MEI}}$	0.306	0.310	0.322

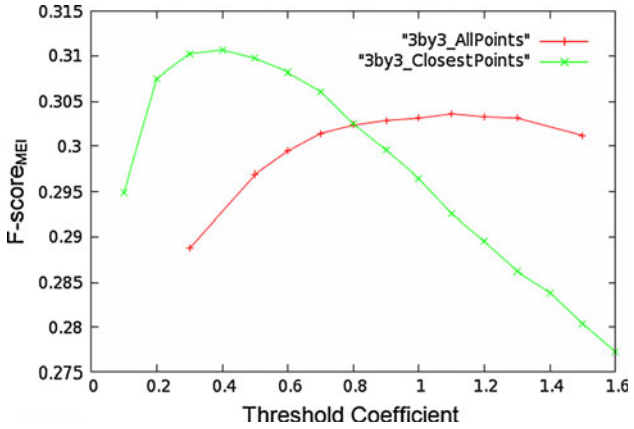

Fig. 12 Results obtained with different point selection techniques, both with marginal distribution

In the following we will test Kernel-based filters on the MEI. To better evaluate the filters, we introduce a new Ground Truth.

So far we used a handmade GT, where the true foreground was marked with rectangular shaped areas. From now on we will restrict these true foreground points to those pixels that are non-zeros on the MEI: the true foreground can be created with a logical AND between the handmade GT and the MEI.

This modification makes sense since our initial data for Kernel-based filtering is the MEI, which means that only the non-zero pixels of the MEI has the chance to be on the final foreground mask. Note that, since the GT is different, the following results are not directly comparable with the results of the previous sub-section (Sec. 6.2.1.). To be unambiguous the F score values, calculated on this modified GT will be named as F $\text{score}_{\text{MEI}}$.

6.3.1 *Patch size*

If only few consecutive frames are available, it is natural to use measurements from the surrounding area of a pixel. It will not only raise the number of measurements but might help dealing with smaller motion compensation errors (see Sec. 4.3). We compared the F $\text{scores}_{\text{MEI}}$ of kernel methods in the case of 1 × 1, 3 × 3 and 5 × 5 sized patches as a function of probability threshold coefficient (20). See Fig. 11.

While in the case of a 1 × 1 patch size the F $\text{score}_{\text{MEI}}$ is more stable, with a larger patch size it has higher peak and it drops very quickly. This can be explained by the following: the threshold values are calculated as a function of average kernel heights. In the case of a larger patch the height of the kernels will be higher, since the measurement values are closer to each other and this results small sigma values. If the threshold is higher, the changes of the coefficient have greater impact on the result.

The test results confirmed that larger patch size is more suitable in our "wearable" case (see Table 2 for summary). For sake of computational complexity we used 3 × 3 sized patches.

6.3.2 *Measurement point selection techniques for joint and marginal representation*

If not only previous pixel values are used as measurements, but the measurement values are selected from a patch, then different methods can be used for selecting points from the patch. Here we compare two ways for measurement selection (see Sec. 4.3). The first is to use all values from the patch, the second is to use only the closest value in the colour space.

Figure 12 shows the results obtained by different point selection techniques in case of a 3 × 3 patch. The corresponding peak F $\text{score}_{\text{MEI}}$ values can be found in Table 3. We can see that selecting only the closest point from a patch gives better results than selecting all points from it. Selecting the closest value helps correcting small errors of motion compensation without adding too much noise to the estimation.

However, using only one point from a patch does not increase the number of measurements, just make the measurements more accurate. In case of marginal PDF calculation it enhances the results, but if we use joint distribution PDF over the colour space we can see that the number of measurements is not enough with respect to the number of dimensions. This explains why the *all value* selection method works better in case of joint distribution, as it can be seen on Fig. 13.

Table 3 The best results obtained with joint and marginal distribution

Distribution	Marginal					Joint		
Patch size	1 × 1	3 × 3		5 × 5		1 × 1	3 × 3	
Measurement selection	N/A	All	Closest	All	Closest	N/A	All	Closest
Peak F $\text{score}_{\text{MEI}}$	0.305	0.303	0.310	not tested	0.322	0.304	**0.342**	0.306

The best value is highlighted with bold

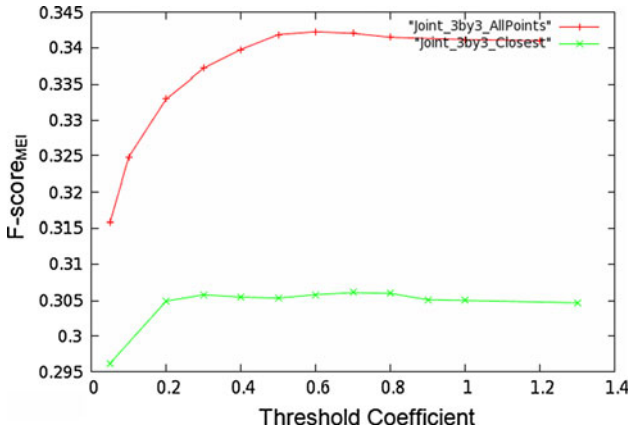

Fig. 13 Results obtained with "all points" and "closest point" selection techniques, both using joint distribution

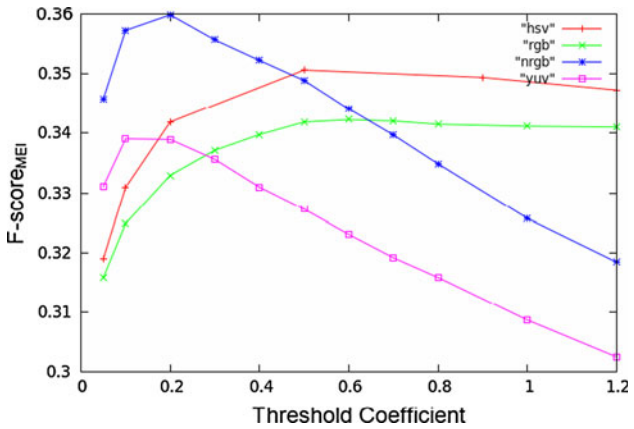

Fig. 14 Results obtained in 3 different colour spaces

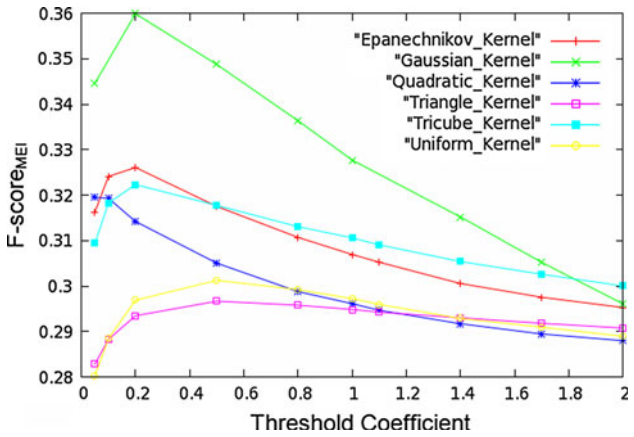

Fig. 15 F-scores$_{\mathrm{MEI}}$ with different Kernel functions as a function of threshold coefficient

Table 3 shows the best results obtained with joint and marginal distribution with optimal patch size and measurement selection method.

6.3.3 *Effect of the choice of the colour space*

We also have examined the performance of the filter in different colour spaces. Figure 14 shows the measurements taken in RGB, normalised RGB, HSV, and YUV colour spaces. We obtained the best results in normalised RGB colour space. Although HSV and RGB colour spaces are more stable, since the threshold coefficient is a chosen constant (see 20), it is more important that the highest value of the curves is better for nRGB colour space.

6.3.4 *Effect of the choice of the kernel function*

The method was tested with different Kernel functions: Gaussian, Quadratic, Tricube, Epanechnikov, Triangle and Uniform kernels. Figure 15 shows the obtained results. We got the best result with Gaussian kernel function, which is the smoothest of the tested kernel functions and this property has high importance in case of a small number of measurements.

6.3.5 *Choice of the kernel width*

Here we compare three methods for kernel width calculation: using a constant value for bandwidth or the distance from kth nearest neighbour or the average distance from the closest k nearest neighbours. Using fixed bandwidth gives significantly lower F score$_{\mathrm{MEI}}$ values than the other two: the peak F score$_{\mathrm{MEI}}$ is 0.302. The other two methods show similar results, however, using the k nearest neighbours gives slightly higher F scores$_{\mathrm{MEI}}$ (see Fig. 16).

6.3.6 *Summary of the chosen parameters*

Table 4 summarises the decisions for the selection of parameters.

6.4 Overall detection performance of the proposed method

The proposed method was compared to a GMM based foreground object detection described in [8] (implementation available at [31]).

Table 5 shows the final results of the proposed method and the alternative method (Gaussian Mixture Model based method) on videos acquired with a standard button camera. The results we obtain are almost 3 times better on these complex sequences than those of the method [8]. Some example results of the compared methods are given in Fig. 17. In these experiments moving objects were shot by a standard camera and the persons were not very close to the device. The ground truth was made by hand for all the sequences.

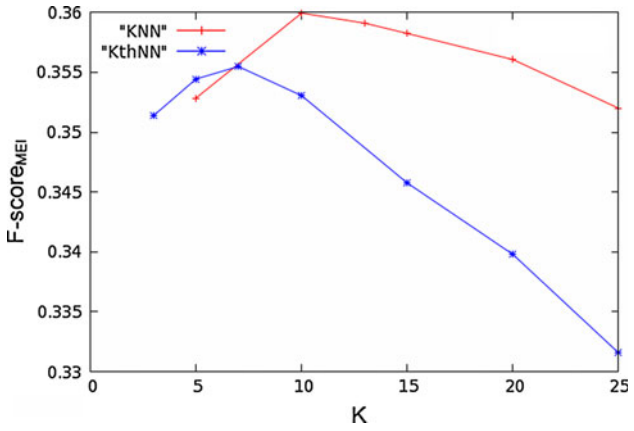

Fig. 16 Comparison of kNN and kthNN bandwidth selection methods

The precision and recall rates were calculated based on the overlap between the pixels annotated as foreground in the ground truth and the estimated foreground image in case of both methods. Both the ground truth and the estimated foreground are rectangular shaped. See the illustration on Fig. 18. Our proposed method performs better both in recall and precision metrics.

6.5 Experiments on "empty" sequences

To assess false detection rate of our method we have also made experiments on "empty" sequence (see Fig. 19) where no moving objects were available. Both methods give false positives, but our proposed method is more than 70% better in average (lower curve in Fig. 19).

6.6 Time performance

The algorithm was tested on Intel Pentim 4, 3.4 GHz CPU, 1 GB memory with Linux operating system. In the present state the run time of the algorithm is far from real-time (see Table 6), since we use non-optimised software implementation without hardware acceleration. This therefore requires offline processing of the recorded data.

Table 5 Precision, recall and *F* score rates for 4 different sequences for the proposed and a concurrent method

Sequence name	# Frames	GMM based method	Proposed method
Francois 1 (indoor)	60	0.050/0.332/**0.087**	0.365/0.906/**0.52**
Francois 2 (indoor)	141	0.242/0.331/**0.279**	0.801/0.799/**0.8**
Daniel 1 (outdoor)	90	0.267/0.428/**0.329**	0.624/0.574/**0.598**
Daniel 2 (outdoor)	30	0.236/0.302/**0.265**	0.467/0.772/**0.582**

The most informative values are highlighted with bold

Since the most time consuming steps are well parallelizable the use of GPGPU can be a perspective for real-time processing. As Table 6 shows, the most time consuming steps are the Block-Matching and the Kernel-Based Foreground Filtering. The parallelization of the former is well studied in the literature [33, 34]. For the latter let us examine its time consumption in detail.

Table 7 shows the computational time needed for processing an average patch. It can be seen that the calculation of one patch takes only a few milliseconds and as the patches are independent from each other a naive way of parallelization is to handle each one of them as an independent thread.

Another approach could be going down to pixel level. According to our measurements the most time demanding step in the processing of a patch is the calculation of the threshold, which is essentially the repeated calculation of (11). Equation (11) is the sum of Gaussian kernel values that can be calculated independently. Hence a promising way of parallelization is the parallel processing of the components of (11). A drawback in this case is the increased number of memory accesses compared to the patch-based decomposition.

These are two ways of breaking down the problem into parallel threads. Other, more sophisticated ways of

Table 4 Summary of the decisions at parameter selection

Density estimation method	Probability representation	Patch size	Measurement pixel selection methods	Colour Space	Kernel function	Kernel width
Mixture of Gaussians	Marginal	1 × 1	**All**	RGB	**Gaussian**	Fix
				nRGB	Epanechnikov	
		3 × 3		HSV	Tricube	Kth Nearest Neighbour
Kernel-based estimation	**Joint**		Closest		Uniform	
		5 × 5		YUV	Triangle	**K Nearest Neighbour**
					Quadratic	

Our choices are marked in bold

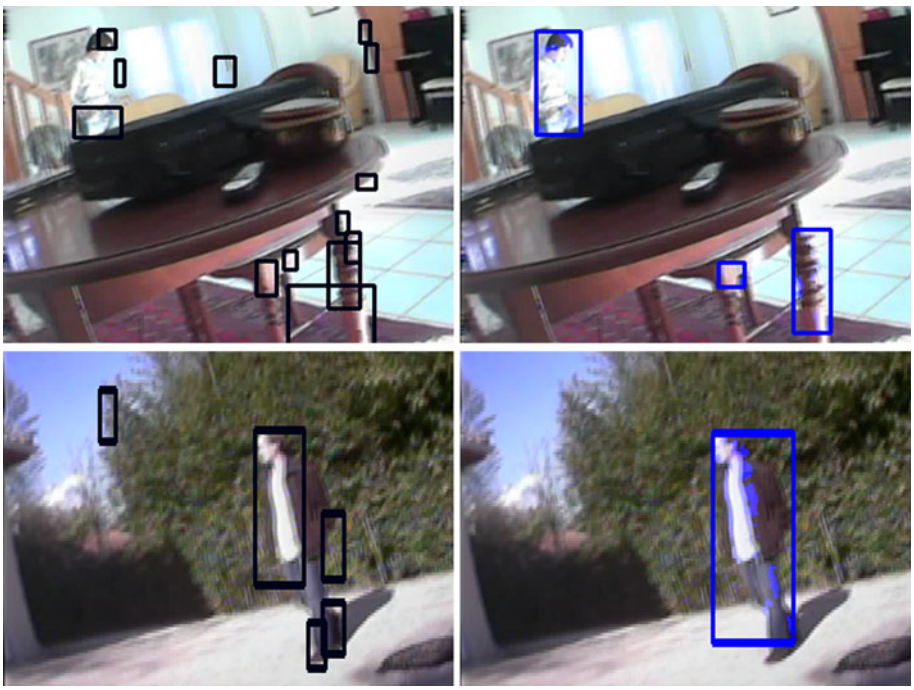

Fig. 17 Example images of foreground detection of the GMM based (*left*) and the proposed method (*right*)

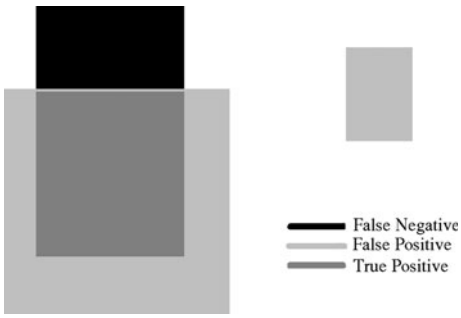

Fig. 18 Illustration of the regions used for evaluation

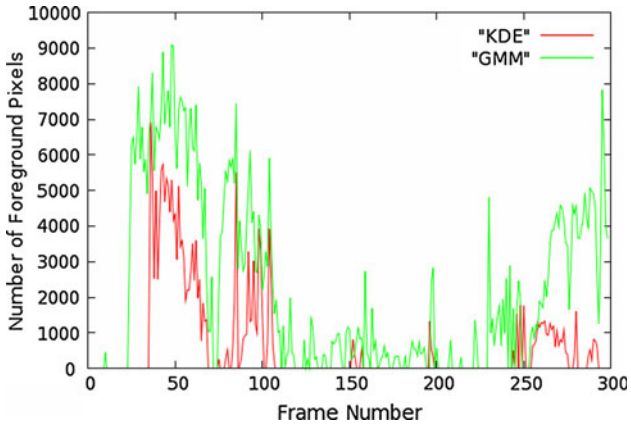

Fig. 19 The number of false foreground pixels on an empty sequence

parallelization may also be studied, but this is not a subject of this paper.

7 Conclusion and perspectives

This work was motivated by recent studies [4] about the correlation of restriction in IADL with the future appearance of a dementia related disease. This result suggests that monitoring of IADL could help in the early detection of dementia. Hence in this paper we proposed a method for detection of moving objects from video frames, recorded with a wearable moving camera carried by patients. We estimated and compensated strong camera motion with a block-matching-based global motion estimator, and used a motion-compensated frame differencing for change detection. To enhance the result of the frame differencing we

Table 6 Time consumption of the main steps of the algorithm in seconds

Block-matching	Motion compensation	Kernel-based foreground filtering	Clustering with DBSCAN	Overall
16.0209	0.1210	23.1958	0.0761	39.4138

Table 7 Time consumption of the Kernel-based foreground filtering of one patch in milliseconds

One kernel (one component of (11)	One PDF value (11)	Bandwidth of a kernel	Threshold	Overall
0.0044	0.0314	0.8918	8.0261	8.9537

proposed a novel, kernel-based PDF foreground filter model to eliminate false detections. Here we followed the approach of K_n nearest neighbours with a small amount of measurements and proposed a novel scheme for the choice of the scale parameter of Gaussian kernels. To detect moving foreground pixels we proposed an adaptive thresholding scheme using heuristics on the shape of estimated PDFs.

On the remaining foreground points the DBSCAN clustering algorithm was used to build foreground objects from the points. It allowed for elimination of isolated noisy detection results thus reducing the false detections.

In the application context of wearable cameras, the problem we addressed was really challenging. We had to estimate PDFs based on the information of 15 frames only while the camera shows the same scene. Beside this strict condition, the quality of many frames was very poor due to the strong motion of the camera.

The time performance is reasonable for offline processing, specifically taking into account the fact that this processing is not necessary for each frame in the recorded video, if a convenient tracking approach is proposed for the detected objects.

Furthermore, for more semantic interpretation of the content, in order to give the medical researchers more insights on the recorded content, learning of object appearance models is necessary. This is a subject of our future work as well.

Finally, to speed up our algorithm we are considering using General-Purpose computing on Graphics Processing Units (GPGPU) [35].

Acknowledgments This work has been supported by French national grant "Eiffel doctorate" and research project PEPS S2TI CNRS "Wearable video monitoring: application to surveillance of persons with age dementia", 2007–2008 and BQR grant of University Bordeaux 1. We also thank Pr. Tamás Szirányi, PPCU, Budapest for fruitful discussion when preparing this paper.

References

1. Mann S (1997) Wearable computing: a first step toward personal imaging. Computer 30(2):25–32
2. Hodges S, Williams L, Berry E, Izadi S, Srinivasan J, Butler A, Smyth G, Kapur N, Wood K (2006) SenseCam: a retrospective memory aid, International Conference on Ubiquitous Computing. LNCS 4206:177–193
3. Personal and ubiquitous computing, special issue on Memory and Sharing of Experiences. (2007) Springer 11(4), pp 213–328
4. Perez K, Helmer C, Amieva H, Orgogozo J-M, Rouch I, Dartigues J-F, Barberger-Gateau P (2008) Natural history of decline in instrumental activities of daily living performance over the 10 years preceding the clinical diagnosis of dementia: a prospective population-based study. J Am Geriatr Soc 56(1):37–44
5. Megret R, Szolgay D, Benois-Pineau J, Joly Ph, Pinquier J, Dartigues J-F, Helmer C (2008) Wearable video monitoring of people with age dementia: video indexing at the service of healthcare. In: Proceedings of the International Workshop on Content-Based Multimedia Indexing, London, pp 101–108
6. Stauffer C, Grimson E (2000) Learning patterns of activity using real-time tracking. IEEE Trans Pattern Anal Mach Intell 22(8): 747–757
7. Zivkovic Z (2004) Improved adaptive Gaussian mixture model for background subtraction. Proc Intern Conf Pattern Recognit 2:28–31
8. Zivkovic Z, van der Heijden F (2006) Efficient adaptive density estimation per image pixel for the task of background subtraction. Pattern Recognit Lett 27(7):773–780
9. Carminati L, Benois-Pineau J (2005) Gaussian mixture classification for moving object detection in video surveillance environment. Proc IEEE Int Conf Image Process 3:113–116
10. Balcells-Capellades M, DeMenthon D, Doermann D (2004) An appearance-based approach for consistent labeling of humans and objects in video. Pattern Anal Appl 7(4):373–385
11. Kim K, Chalidabhongse TH, Harwood D, Davis L (2004) Background modeling and subtraction by codebook construction. Proceedings of IEEE International Conference of Image Processing, vol 5, pp 3061–3064
12. Mittal A, Paragios N (2004) Motion-based background subtraction using adaptive kernel density estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 302–309
13. Tiand T, Tomasi C, Heeger D (1996) Comparision of approaches to ego-motion computation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 315–320
14. Foresti GL, Micheloni C (2003) A robust feature tracker for active surveillance of outdoor scenes. Electron Lett Comput Vis Image Anal 1(1):21–34
15. Jung B, Sukhatme GS (2004) Detecting moving objects using a single camera on a mobile robot in an outdoor environment. In: the Conference on Intelligent Autonomous Systems, Amsterdam, pp 980–987
16. Jung-Ho A, Cheolmin C, Sooyeong K, Kilcheon K, Hyeran B (2008) Human tracking and silhouette extraction for human–robot interaction systems. Pattern Anal Appl (published online)
17. Veit T, Cao F, Bouthemy (2006) An a contrario decision framework for region-based motion detection. Intern J Comput Vis 68(2):163–178
18. Yuan C, Medioni G, Kang J, Cohen I (2007) Detecting motion regions in the presence of a strong parallax from a moving camera by multiview geometric constraints. IEEE Trans Pattern Anal Mach Intell 29(9):1627–1641
19. Sheikh Y, Shah M (2005) Bayesian modeling of dynamic scenes for object detection. IEEE Trans Pattern Anal Mach Intell 27(11):1778–1792
20. Bierling M (1988) Displacement estimation by hierarchical block matching. Proc SPIE Vis Commun Image Process 1001:942–951
21. Accame M, de Natale FGB, Giusto DD (1998) High performance hierarchical block-based motion estimation for real-time video coding. Real-Time Imaging 4(1):67–79
22. Durik M, Benois-Pineau J (2001) Robust motion characterisation for video indexing based on MPEG2 opticalflow. In: Proceedings of the International Workshop on Content-Based Multimedia Indexing, pp 57–64
23. Parzen E (1962) On estimation of a probability density function and mode. Ann Math Stat 33:1065–1076
24. Archambeau C, Valle M, Assenza A, Verleysen M (2006) Assessment of probability density estimation methods: Parzen window and finite Gaussian mixtures. In: Proceedings of IEEE International Symposium on Circuits and Systems, pp 1–4
25. Duda RO, Hart PE, Stork DG (2001) Pattern classification, 2nd edn. John Wiley & Sons, Inc., NY

26. Bugeau A (2007) Détection et suivi d'objets en mouvement dans des scènes complexes, application à la surveillance des conducteurs, Thèse de l'université de Rennes 1, Mention Traitement du Signal et des Télécommunications
27. Epanechnikov (1969) Nonparametric estimates of a multivariate probability density. Theor Probab Appl 14:153–158
28. M. Ester, H.-P. Kriegel, J. Sander, X. Xu (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of Second International Conference on Knowledge Discovery and Data Mining. Portland, OR, pp 226–231
29. Sander J, Ester M, Kriegel H-P, Xu X (1998) Density-based clustering in spatial databases: the algorithm GDBSCAN and its applications. Data Min Knowl Disc 2:169–194
30. van Rijsbergen CJ (1979) Information retrieval, 2nd edn. Butterworth-Heinemann, London
31. http://staff.science.uva.nl/~zivkovic/DOWNLOAD.html
32. Zivkovic Z, van der Heijden F (2004) Recursive unsupervised learning of finite mixture models. IEEE Trans Pattern Anal Mach Intell 26(5):651–656
33. Gupta G, Chakrabarti C (1995) Architectures for hierarchical and other block matching algorithm. IEEE Trans Circuits Sys Video Technol 5:477–489
34. Mazaré S, Pacalet R, Dugelay J-L (2006) Using GPU for fast block-matching. In: 14th European Signal Processing Conference, Florence, Italy
35. Owens JD, Luebke D, Govindaraju N, Harris M, Krüger J, Lefohn AE, Purcell T (2007) A survey of general-purpose computation on graphics hardware. Comput Graph Forum 26(1): 80–113