A Fast Approach to Novelty Detection in Video Streams using Recursive Density Estimation

Ramin Ramezani, Student Member IEEE, Plamen Angelov, Senior Member IEEE, Xiaowei Zhou, Student Member IEEE

Abstract — Video-based surveillance and security become extremely important in the new, 21st century for human safety, counter-terrorism, traffic control etc. Visual novelty detection and tracking are key elements of such activities. The current state-of-the-art approaches often suffer from high computational, memory storage costs and from not being fully automated (they usually require a human operator in the loop). This paper introduces a new approach to the problem of novelty detection in video streams that is based on recursive, and therefore, computationally efficient density estimation by a Cauchy type of kernel (as opposed to the usually used Gaussian one). The idea of the proposed approach stems from the recently introduced evolving clustering approach, eClustering and is suitable for on-line and real-time applications in fully autonomous and unsupervised systems as a stand-alone novelty detector or for priming a tracking algorithm. The approach proposed in this paper has evolving property - it can gradually update the background model and the criteria to detect novelty by unsupervised on-line learning. The proposed approach is faster by an order of magnitude than the well known kernel density estimation (KDE) method for background subtraction, while having adaptive characteristics, and does not need any threshold to be pre-specified. Recursive expressions similar to the proposed approach in this paper can also be applied to image segmentation and landmark recognition used for self-localization in robotics. If combined with a real-time prediction using Kalman filter or evolving Takagi-Sugeno fuzzy models a fast and fully autonomous tracking system can be realized with potential applications in surveillance and robotic systems.

I. INTRODUCTION

The key function of security and surveillance systems is to enable the user to keep track of activities that take place in the environment that is being monitored [1,3-7]. Most of these systems make use of optical, infrared cameras or 3D laser scanners which results in images and video streams that can be processed on-line, transmitted or recorded and stored on a disk for further off-line processing (usually involving human operators) [2]. The main challenges are to develop fully automatic systems that require limited processing time and storage capacity and are thus applicable on-line and in real-time; that does not require task-specific thresholds and tuning. This underlines the importance of algorithms that are computationally efficient, task-, operator-, and threshold-independent and are capable of detecting and tracking activities in a scene of observation.

In certain circumstances like unknown environments, human interaction is risky; therefore, developing algorithms to facilitate machines to execute certain tasks without having prior knowledge about the environment is of a vital importance. In a security monitoring scenario such ability will remove the problem of boredom and lack of concentration of the operators.

Traditionally, in visual surveillance systems, huge volumes of data are routinely stored in the form of image frames in order to be inspected by humans after events happening on the scene observed by a camera. These off-line methods require a large amount of computer storage for archiving video streams and hence are not very efficient [1,2]. The most prominent approaches are based on so called background subtraction and background modelling [1,3,4,6] and will be reviewed in the next section.

On-line algorithms could facilitate the interpretations of video sequences in real time as well as reducing the storage requirement for archiving the videos. These methods could also enhance the video transmission by discarding the unnecessary data and keeping the valuable information about those particular objects classified as important ones; besides unsupervised algorithms eliminate the possibility of potential mistakes by human operators.

The proposed approach stems from the well known Kernel Density Estimation (KDE) procedure [1] which is described in more detail in the next section. The main idea of the proposed approach is to approximate the probability distribution function (pdf) of the colour intensity by a Cauchy type kernel (instead of the Gaussian kernel used in the original KDE approach) and, after that, to use a recursive expression to update this estimation on-line by the information from the pixel colour intensity that the next image frame brings. In this way, there is no need to keep in the memory previous image frames. Instead accumulated information that represents the colour density per pixel is kept in the memory only, which has dimensions of 3 frames for colour images (one frame size for each of the basic colours – red, R; green, G; and blue, B; or alternatively one for each of hue, H; saturation, S; and brightness value, V) and one frame for grey images. Contrast this to the necessity to keep in the memory N different frames, where $N$ is the size of a window (usually $N>10$). In this way, if the size of the frame is $M$ pixels, the memory requirements of KDE approach are $3\times N \times M$ values for colour and $N \times M$ values for grey images.
for grey images to be stored while the memory requirements for the proposed approach are 3xM or M respectively. In this way, the proposed recursive density estimation (RDE) approach requires N times less memory (where N is usually >10). The processing time is also reduced proportionally. A comparison of the processing time and storage required by the original KDE and the proposed RDE approaches is given in the experimental results section.

The proposed algorithm takes its roots from the recently introduced evolving clustering procedure, eClustering which is a real – time algorithm that is threshold – free. The main idea of recursive Cauchy – type density estimation is used here to cluster the pixel colour intensity online into background and foreground (pixels for which any significant novelty is detected). This approach can be extended for automatic object tracking when combined with Kalman filter or evolving Takagi – Sugeno fuzzy model in a similar way as in [7], for image segmentation [8] and landmark detection [9] used in self – localisation in robotics [10]. Another practical application is to use such a system to reduce the huge amount of data and to automatically select suspicious areas from a scene of observation. In such a scenario, a surveillance and patrol system can automatically focus the attention of the scene of observation. In such a scenario, a surveillance and patrol system can automatically focus the attention of the scene of observation.

II. NOVELTY DETECTION IN VIDEO STREAMS THROUGH BACKGROUND SUBTRACTION: A REVIEW

A. Background subtraction

Background subtraction is one of the most popular methods for novelty detection in video streams. It focuses on two major steps; first, to construct a statistical representation of the background that is representative, robust to the noise and is sensitive to new objects, and second, to build another statistical model called ‘foreground’ that represents the changes that take place on the scene. By applying this approach to each frame one effectively achieves tracking any moving object. Robustness is required to cope with noise due to environmental fluctuations such as wind, movement of tree leaves etc. that are not due to the appearance of a new object on the scene.

A method for building background or foreground models is known as statistical modeling, where each pixel in an image is modeled as a random variable in a particular feature space along with the probability density function (pdf).

Frame Differencing plays an important role in background subtraction depicting the general notion of object tracking. In this way, all pixels of the current frame are compared with the respective pixels from the previous frames and each pixel which shows a major variation in its value is assumed to be a foreground. In some works [3] the absolute difference between every two frames is calculated and using a threshold the decision is taken. The deficiency of this method is the sensitivity to threshold and low robustness to illumination changes and camera oscillations.

Another method is called mixture of Gaussians [21]. In order to estimate pdf of the background in most of the method using Gaussian distribution it is needed to fit Gaussian distribution for every frame and update it through time. However, in mixture of Gaussians, multimodal background distribution is assumed. All distributions are specified based on their weights and distances matching the current value of a pixel at any particular position. Drawback of this model is also threshold dependency and the decision making required for selecting the proper distribution as a background model.

B. Kernel Density Estimation

Kernel Density Estimation (KDE) is one of the most common techniques introduced for modeling the background in video stream processing. KDE approach assumes a Gaussian kernel to represent the pdf of each pixel [1,6]. The probability of a pixel colour intensity value to be a part of the background can be estimated numerically from training data based on a window with length N (usually N>10 [1] but not excessively large because the background is not necessarily the same for the whole video stream). In this way, the probability of a pixel to represent background can be estimated and in case of being lower than a pre-specified threshold, this pixel is assumed to be a foreground (a visual novelty – a part of a new object that appears on the scene in the current frame). The threshold is usually selected subjectively or as a result of a number of off-line experiments [1].

KDE is very accurate approach [1,6], but nevertheless it is very expensive memory-wise and computation-wise due to the need to store the previous N frames. Additionally, a suitable value of the threshold must be found (usually experimentally and off-line).

In novelty detection and object tracking, background (or normality) is a relative concept and is determined by comparing a number (N) of consecutive frames. Besides certain portions of an image which is specified as foreground could once be a part of background and vice versa. Imagine a car enters an image and then stops; during this event, portions of the image are first background then converting to foreground and later they will be allocated as background again relative to future frames. Another note which should be mentioned is that, no background or foreground model could be attained right away after the first frame. The usual concept relies on using a window (buffer) of certain size (N) in an off-line fashion.

Let x(1), x(2), ....x(t),....x(N) be a sample group of colour intensity values for N consecutive frames of a video stream for certain pixel position in each frame as illustrated in Figure 1. The pdf of the current, \( i^{th} \) pixel, \( x(t) \) to be a part of the background is estimated in the KDE approach [1] as an average similarity of its colour intensity value to all colour
cases;
Which is equivalent to:
The most commonly used kernel function is Gaussian which
leads to [1]:
The intensity values of pixels in the same position in a window of
$N$ frames by:
$$p(x(t)) = \frac{1}{N} \sum_{i=1}^{N} \prod_{j=1}^{r} k_{\sigma}(x(t) - x(i))$$
where $x(i)$ denotes the colour intensity value of the pixel in
$i^{th}$ frame; $k_{\sigma}$ is the kernel function with bandwidth, $\sigma$ [1].

Equation (1) assumes one dimensional representation of the
colour intensity value which is enough in grey image case. If
use colour image a 3D representation (for R,G,B or H,S,V [2])
is necessary. In such cases the equation (1) generalizes to [1]:
$$p(x(t)) = \frac{1}{N} \sum_{i=1}^{N} \prod_{j=1}^{r} k_{\sigma}(x(t) - x(i))$$

where $r$ denotes the color channel and is equal to 3 in most
cases; $\sigma$ is the bandwidth of the kernel function in that
particular channel.

This formula shows that the probability of every pixel’s
intensity in frame $t$ is the average product of kernel functions.
The most commonly used kernel function is Gaussian which
leads to [1]:
$$p(x(t)) = \frac{1}{N} \sum_{i=1}^{N} \prod_{j=1}^{r} \frac{1}{\sqrt{2\pi\sigma_j^2}} e^{-\frac{(x(t) - x(i))^2}{2\sigma_j^2}}$$

(3)

Which is equivalent to:
$$p(x(t)) = \frac{1}{N} \sum_{i=1}^{N} q \sum_{m} \frac{x(i) - x(m)}{2\sigma_j^2}$$

(3a)

Where $q = \frac{1}{\sqrt{2\pi\sigma_j^2}}$

After estimating the probability, the following condition is
checked:
IF $(p(x(t))< threshold)$
THEN $(x(t)$ is $FG)$
ELSE $(x(t)$ is $BG)$

Where $FG$ denotes the foreground
$BG$ denotes background

The threshold is assigned globally and could be adjusted in
order to lower the percentage of false detection. Practically,
the probability in (3) is estimated by calculating the kernel
function in offline mode as a lookup table based on the
difference of intensity values. In this method the estimation
takes place using a window function as most recent samples
which means adding new frames and discarding the previous
ones [1]. Note that defining a proper threshold is a significant
drawback and may result in distortion and low performance of
the whole system in different environments. The proposed
new RDE algorithm overcomes this problem and the problem
of off-line computation of the estimate of the pdf. Another
major concern for this type of estimation is to define a proper
bandwidth for the kernel function. Theoretically, as the
number of samples increases to infinity, the role of the
bandwidth fades. The choice of a suitable bandwidth plays an
important role in KDE as in case of a narrow bandwidth,
density estimation will tend to be ragged; on the contrary, a
wide bandwidth causes the density estimation to be
over-smoothed [1]. Defining the proper bandwidth is also
another drawback of this method. This deficiency along with
the high memory usage of this method is reduced by the
proposed RDE approach which is using recursive calculations
instead which still take into account $N$ frames, but does not
require storing them in the memory or manipulating them
off-line. Instead they are processed on-line in one pass.

III. RECURSIVE DENSITY ESTIMATION (RDE) APPROACH

A. The concept

The main drivers in the proposed approach are the attempt
to estimate a density function that approximates the pdf given
by equation (1)-(3) in a recursive fashion and to eliminate the
dependence of subjective threshold and parameters (such as
the bandwidth). First, we noted that the KDE by (1)-(3) is a
special case of the more generic Parzen windows [12],
generalized regression models [13], and Mountain function
[14] and potential [15] used in respective clustering
algorithms. In a similar manner as in the so called eClustering
algorithm [16] the Gaussian kernel can be approximated by a
Cauchy function because the Cauchy function has the same
basic properties as the Gaussian [16]: i) it is monotonic; ii) its
maximum is unique and of value 1; iii) it asymptotically tends
to zero when the argument tends to plus or minus infinity. In
fact, Cauchy function is a first order approximation of the
Gaussian in Taylor series:

$$f^{(1)}(x) e^{-\frac{(x(t) - x(i))^2}{2\sigma_j^2}} = \frac{1}{e^{\pi/\sigma_j^2}} \left(\frac{1}{\sigma_j^2} \sum_{i=1}^{N} \frac{(x(t) - x(i))^2}{2\sigma_j^2} + ...\right)$$

(5)
The data density has been used as a criterion to form new clusters in the so-called evolving fuzzy clustering approach, eClustering [16] forming so-called potential. Potential is an estimate of the importance of a data sample (pixel color intensity value) based on the density and similarity to all previously seen data samples or samples from a window with length $L$ [17]:

$$P(x(t)) = \frac{1}{1 + \sum_{j=1}^{r} \sum_{i=1}^{L} \frac{(x_j(t) - x_i(t))^2}{2\sigma_j^2}}$$

(6)

B. Recursive Estimation of the Density by a Cauchy – type of kernel

Potential of each data sample (particular pixel in the current image frame represented with $r$ color channels) is the function of accumulated distance between this data sample and all other data samples (pixels with the same position in previous $N$ image frames). Because this operation needs to be applied on every single pixel (same as it is in the original KDE expression that calculates the potential as expressed in (6)) has the aim of the proposed RDE approach. The recursive accumulating values that will allow a recursive estimation of the color density information as represented by (6). This is the purpose to process images frame by frame (on-line) and discard the image (which usually is a large number). This also increases the time of processing these $N$ frames. An alternative is to process images frame by frame (on-line) and discard the frames that have been processed already, but still taking into account the visual information that they contain by accumulating values that will allow a recursive estimation of the color density information as represented by (6). This is the aim of the proposed RDE approach. The recursive expression that calculates the potential as expressed in (6) has been derived elsewhere and we recommend the reader to visit these references for more details [16, 19]:

$$P(x(t)) = \frac{t-1}{(t-1)(a(t)+1)+b(t)-2c(t)}$$

(7)

Values $a(t)$ and $c(t)$ can be calculated from the current frame only:

$$a(t) = \sum_{j=1}^{r} x_j^2(t)$$

(8)

$$c(t) = \sum_{j=1}^{r} x_j(t)d_j(t)$$

(9)

Where $d_j(t)$ is calculated recursively as shown below.

The value $b(t)$ is also accumulated during the processing of the frames one by one as given by the following recursive expressions:

$$b(t) = b(t-1) + a(t-1)$$

(10a)

$$d_j(t) = d_j(t-1) + x_j(t-1)$$

(11a)

Both quantities are initialized by zeroes [16, 19]:

$$b(1) = 0$$

$$d_j(1) = 0$$

The value of the spread of the Cauchy function $\sigma$ can be updated by the data samples to learn the data variance as described in [20]:

$$\sigma_j^2(t) = \frac{\sigma_j^2(t-1) + \frac{1}{2} \sum_{i=1}^{l_i} (x_i(t) - x_i(t))^2}{2}$$

$$\sigma_j(1) = 0.5$$

(12)

C. Novelty Detection using RDE

The proposed algorithm has two phases. In the first phase the current frame is being processed pixel by pixel (note that a parallel implementation of the approach is possible to reduce further the computational load and realize a real-time implementation). Using the equations (7)-(11) it is possible to estimate the data density in the previous $N$ image frames without keeping them in the memory (working with the pixel from the current frame, $x(t)$ plus the accumulated quantities $b(t)$ and $d(t)$ only). One can also keep track of the minimum value of the potential met so far (let us denote this by $P_{\min}(t)$):

$$P_{\min}(t) = \min_{i=1}^{N} \{P(i)\}$$

(13)

The value of $P_{\min}(t)$ is used in RDE approach as an adaptive, problem-, and user-independent threshold to evaluate whether the current pixel is significantly different from the pixels at the same position in all the previous image frames (in a window). In this way, the condition that is being checked can be formulated as:

$$IF \ (P(x(t)) < P_{\min}(t))$$

THEN $x(t)$ is FG

ELSE $x(t)$ is BG

(14)

A pixel could be in background for a certain period and turns to be foreground and ultimately again background or vice versa while being in the same window. This situation happens when an object enters a part of a scene and before the window comes to an end, it goes out of that portion. In this case the pixel intensity in the last status is equal or similar to its initial value and hence should be allocated to the same cluster. As mentioned before, at any certain time where a pixel changes its label, the status of that pixel should be modified to the next condition considering that there are only two possible conditions either foreground or background.

In phase 2 of the RDE approach one need to determine the shape of the object that has already been detected or will be tracked. Note that this phase also exists in the original KDE approach whereas another threshold is used. Instead, at the second stage of the RDE approach we propose to use again potential (but this time in terms of the spatial position of the pixel in the current frame) instead of the mean value:
\[
P(h_j, v_j) = \frac{1}{1 + \sum_{j=1}^{F} \frac{(h_j - h_j)^2 + (v_j - v_j)^2}{2\sigma_j^2}}
\]

where \( h \) denotes the horizontal position of the pixel in the image frame; \( v \) denotes the vertical position of the pixel in the image frame.

Again, the spatial potential can be calculated recursively similarly to (7)-(11):

\[
P(h_j, v_j) = \frac{j-1}{(j-1)(\alpha(j) + 1) + \beta(j) - 2\gamma(j)}
\]

\[
\alpha(j) = h_j^2 + v_j^2
\]

\[
\gamma(j) = h_j\delta_j(j) + v_j\delta_j(j)
\]

\[
\beta(j) = \beta(j-1) + \alpha(j-1); \beta(1) = 0
\]

\[
\delta_j(j) = \delta_j(j-1) + h_{j-1}; \delta_j(1) = 0
\]

\[
\delta_j(j) = \delta_j(j-1) + v_{j-1}; \delta_j(1) = 0
\]

\[
[l, v_j] = \arg \max_{i=1}^{F} P(i)
\]

where \( \text{Centre} = [h_l, v_l] \) is the centre of the object being detected (and possibly tracked); \( F \) denotes the number of pixels in a frame classified as foreground \((F<M)\).

The pixel with the highest value of potential (calculated in terms of spatial position in the frame!) \( \text{Centre} \) will be the focal point of the object that is surrounded by more pixels classified as foreground (belonging to this novelty object) while spatial mean is prone to the influence by pixels that are classified as foreground but might be due to the noise, small movements of other objects such as leaves of the trees etc. The proposed potential-based way to locate the novelty is more robust if compare to the original KDE approach [1].

D. Advantages of the Proposed RDE Algorithm:

- The original KDE approach is very sensitive to the threshold which should be defined based on experience and training the system for every environment. On the contrary, the proposed RDE method does not need any threshold and does not need to be tuned in different environments.
- The proposed algorithm is faster (in an order of magnitude) that the original KDE approach and requires \( N \) times less memory – see Table I.

Table I. Experimental results using ‘Rain’ video available at [23]

<table>
<thead>
<tr>
<th>Method</th>
<th>Time per frame (seconds)</th>
<th>Frame Size (pixels)</th>
<th>Software</th>
<th>Computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed RDE</td>
<td>0.74723</td>
<td>235x225</td>
<td>MATLAB</td>
<td>Pentium 4, 2.8 GHz, 1 GB of RAM</td>
</tr>
<tr>
<td>KDE</td>
<td>3.72003</td>
<td>235x225</td>
<td>MATLAB</td>
<td>Pentium 4, 2.8 GHz, 1 GB of RAM</td>
</tr>
</tbody>
</table>

Note that the time when using C language instead of Matlab can be significantly reduced and eClustering was realised on hardware (FPGA) [22] which paves the way to real-time implementations.

- RDE requires much less memory \((N \text{ times less})\) comparing to the traditional KDE technique;
- Real-time applications are much more realistic for RDE, while for KDE it is limited not only by the computing power (speed of the processor, RAM) but also methodologically, by the size of the window. Note that if the window (buffer) that is being used in KDE approach is too wide the sensitivity of the approach diminishes. On the contrary, if the window is too short it may lead to an oversensitive realization.
- In a practical realization of the original KDE approach one can use a ‘jumping’ or a ‘sliding’ window; ‘Jumping window lowers the accuracy of the KDE approach but adds to the agility of the overall system. The proposed RDE approach can use a jumping window (it can also use all the available frames so far without memorizing them as well) without deterioration of the performance due to the accumulation of the density estimations in the recursive expressions. Therefore in RDE approach larger window can be used comparing to KDE (see Table I).

Fig. 2. Background Subtraction using the proposed RDE method. Left hand side scenes are original frames; right hand side scenes are modeled ones. The red square denotes the focal point of the foreground.

E. Illumination Changes and Camera Oscillation:

The brightness of a scene may vary from one frame to another, for instance, in an outdoor environment in latter frames clouds may appear in the environment that makes the frames darker than the previous one. Even though clouds most probably may not turn up suddenly in a frame, still they will alter the brightness of the image gradually. On the contrary, some events, may suddenly change the environment illumination, say a car with lights on in a dark environment coming to a scene, lights will brighten a part of the background which used to be dark in the previous frame.
The static camera may oscillate due to terrestrial conditions. Mostly, the cameras are mounted on a shaft or the wall which may face slight shocks because of wind, rain or fast movement of a heavy truck. Consequently, due to this phenomenon, particular pixels do not have the same intensity at the corresponding positions in next frames. Frames are shifted after fluctuation.

Such problems may cause the system to determine more pixels as foreground absurdly which influences proper object tracking, this deficiency could be overcome by stage 2.

As mentioned earlier, mean value of the foreground pixels is assumed as the center of the target whereas after camera oscillations or illumination changes more disperse pixels will appear in the scene, however, if the foreground pixels are supposed to form an identical cluster, by measuring the potential of these pixels, the highest potential for being the center of the cluster represents the maximum likelihood center of the target.

IV. ACKNOWLEDGEMENTS

The authors are grateful to Dr. L. Mihaylova who pointed out some useful references on background subtraction and KDE.

V. CONCLUSION

A new approach to density-estimation for background subtraction applied for novelty detection and tracking in video streams introduced in this paper which is threshold-free and recursive (thus computationally efficient in terms of memory used and time consumed). The key innovation comparing to the KDE approach used in the literature is the new proposal for density estimation by a Cauchy type of kernel (as opposed to the usually used Gaussian one). The idea of the proposed approach stems from the recently introduced evolving clustering approach, eClustering and is suitable for on-line and real-time applications in fully autonomous and unsupervised systems as a stand-alone novelty detector or for priming a tracking algorithm. The approach proposed in this paper has evolving property - it can gradually update the background model and the criteria to detect novelty by unsupervised on-line learning. The proposed approach is faster by an order of magnitude (10 times or more) than the well known kernel density estimation (KDE) method for background subtraction, has adaptive characteristics, and does not need any threshold to be pre-specified. Recursive expressions similar to the proposed approach in this paper can also be applied to image segmentation and landmark recognition used for self-localization in robotics. If combined with a real-time prediction using Kalman filter or evolving Takagi-Sugeno fuzzy models a fast and fully autonomous tracking system can be realized with potential applications in surveillance and robotic systems.

REFERENCES


Date of access: 1st June 2007.