MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Learning a Sparse, Corner-Based Representation for Background Modelling

Qiang Zhu, Shai Avidan, Kwang-Ting Cheng

TR2005-148 October 2005

Abstract

Time-varying phenomenon, such as ripples on water, trees waving in the wind and illumination changes, produces false motions, which significantly compromises the performance of an outdoor-surveillance system. In this paper, we propose a corner-based background model to effectively detect moving-objects in challenging dynamic scenes. Specifically, the method follows a three-step process. First, we detect feature points using a Harris corner detector and represent them as SIFT-like descriptors. Second, we dynamically learn a background model and classify each extracted feature as either a background or a foreground feature. Last, a Lucas-Kanade feature tracker is integrated into this framework to differentiate motion consistent foreground objects from background objects withrandom or repetitive motion. The key insight of our work is that a collection of SIFT-like features can effectively represent the environment and account for variations caused by natural effects with dynamic movements. Features that do not correspond to the background must therefore correspond to foreground moving objects. Our method is computational efficient and works in real-time. Experiments on challenging video clips demonstrate that the proposed method achieves a higher accuracy in detecting the foreground objects than the existing methods.

IEEE International Conference on Computer Vision (ICCV)

Learning a Sparse, Corner-Based Representation for Background Modelling

Qiang Zhu[1], Shai Avidan[2], Kwang-Ting Cheng[1]

[1]Electrical & Computer Engineering Department
University of California at Santa Barbara, CA, 93106, USA
[2]Mitsubishi Electric Research Laboratories
201 Broadway, Cambridge, MA, 02139, USA
{qzhu,timcheng}@ece.ucsb.edu, avidan@merl.com

Abstract

Time-varying phenomenon, such as ripples on water, trees waving in the wind and illumination changes, produces false motions, which significantly compromises the performance of an outdoor-surveillance system. In this paper, we propose a corner-based background model to effectively detect moving-objects in challenging dynamic scenes. Specifically, the method follows a three-step process. First, we detect feature points using a Harris corner detector and represent them as SIFT-like descriptors. Second, we dynamically learn a background model and classify each extracted feature as either a background or a foreground feature. Last, a "$Lucas$-$Kanade$" feature tracker is integrated into this framework to differentiate motion-consistent foreground objects from background objects with random or repetitive motion. The key insight of our work is that a collection of SIFT-like features can effectively represent the environment and account for variations caused by natural effects with dynamic movements. Features that do not correspond to the background must therefore correspond to foreground moving objects. Our method is computational efficient and works in real-time. Experiments on challenging video clips demonstrate that the proposed method achieves a higher accuracy in detecting the foreground objects than the existing methods.

1 Introduction

Surveillance systems seek to automatically and robustly identify pedestrians, vehicles, or events of interest in various environments. With the assumption of a stationary background which allows the use of statistical techniques for background modelling, interesting moving-objects can be distinguished from the background effectively for applications and environments that meet this assumption. However, the assumption of a stationary background can be challenged in many outdoor surveillance scenarios, such as ripples on water, trees waving in the wind and illumination changes.

1.1 Related work

Over time, the intensity value of an individual pixel with a static background usually follows a normal distribution. Hence, a reasonable model to represent such a statistical distribution is a single Gaussian model [11]. However, a single Gaussian is often inadequate to accurately model the temporal changes of a pixel value in dynamic backgrounds, such as changes in light and shadow. The use of multiple hypotheses to describe the behavior of such dynamic scenes at the pixel level was a breakthrough in background modelling. Specifically, methods employing a mixture of Gaussians have become a popular basis for a large number of related techniques in recent years. Friedman [4] proposed a mixture of three Gaussian components to model the visual properties of each pixel, in which Expectation-Maximization (EM) algorithm is proposed to learn such a Gaussian Mixture Model (GMM). In [5], the authors discussed modelling each pixel as a mixture of Gaussians with flexible numbers of Gaussian components, and using an on-line approximation to update the model. Their real-time video surveillance system has been proven robust for day/night cycles and for scene changes over long periods of time. However, for backgrounds exhibiting very rapid variations, such as ripples on water, ocean waves, or moving trees, Gaussian Mixture Model can result in a distribution with large variance over long video sequences, thus significantly reducing the sensitivity for detecting foreground objects. To address the dynamic backgrounds, non-parametric method has recently been developed which use kernel density estimation technique to predict background-pixel value based on multiple recently collected samples. This technique can adapt very promptly to

rapid background changes. In [3], the authors introduced a novel non-parametric background model and a background-subtraction approach. Their method uses a normal kernel function for density estimation. The learned model represents a history of recent sample values over a long sequence, and it adapts to the background changes quickly. A similar approach was described in [9], which emphasizes more of a variable bandwidth kernel for the purpose of adaptive density estimation.

Other efforts dealing with background modelling can be categorized as predictive methods, which treat pixel value changes as a time series and use a temporal model to predict the next pixel value based on past observations. Any deviation between the predicted value and the actual observation can be used to adjust the predictive model parameters. In [12], an autoregressive model was proposed to capture the properties of dynamic scenes for the purpose of foreground detection in video surveillance.

1.2 Our corner-based approach

In general, pixel-level background modelling suffers from two major disadvantages. First, the computational complexity of these methods is inherently high, since every pixel must be processed in each video frame. In many challenging dynamic scenes, a number of different frequency components demand a model consisting of many Gaussians, or a highly complicated predictive model to precisely capture the recurrent patterns of motion at a single pixel over time. The performance tradeoff between detection accuracy and computation cost is always a hard decision in choosing a pixel-level framework. Secondly, the intensity value at individual pixels is very easily affected by image noise and does not fully exploit the useful correlation of the spatially neighboring pixels. In essence, what is lacking in such approaches is some higher level information, which is more robust and can be derived from regions in the image or even from the entire frame.

We were inspired by recent advances in feature-based object representation. For example, Lowe [2] uses a collection of SIFT descriptors to represent an object. At the other end of the spectrum, the $ASSET$-2 system [10] uses feature points to detect and track moving objects such as vehicles observed by a moving camera. Our objective in this work is to use a set of SIFT-like features to model the entire scene, instead of foreground moving-objects. The robustness of SIFT-like descriptors could tolerate background variations caused by natural phenomena such as ripples on the water, swaying trees or illumination changes. Another unique idea behind our method is that SIFT-like descriptors are used as a strong cue for detecting and matching feature points across video frames. In comparison, the $ASSET$-2 system models each feature point in a very simple representation using the smoothed image brightness and the **x, y** image derivatives. In addition, their detection and tracking of moving objects relies heavily on clustering results of feature points with the same motion.

In this paper, we propose a novel modelling technique, which is based on a sparse feature set of detected corners in each video frame. The proposed feature-based approach follows a three-step process. For every video frame, we detect the corners using a Harris corner-detector, and then, we describe and represent them as SIFT-like features. Based on this feature set, we build and maintain a dynamic background model which is able to account for variations caused by natural dynamic effects. Using the learned model, we classify each feature into either a background or a foreground feature. In the last step, we propose a "$Lucas$-$Kanade$" feature tracker to track each foreground feature over time, where a temporally and spatially coherent cluster of the tracked features indicates a real moving-object.

The remainder of the paper is organized as follows. Section 2 describes the corner detection and represents each detected corner as a SIFT-like feature. In Section 3, we introduce a corner-based background model, and by which we classify each feature as either a foreground or a background feature. Section 4 further improves the detection accuracy of the real moving-objects using a "$Lucas$-$Kanade$" feature tracker. Experimental results will be analyzed in Section 5. In the last Section, we conclude with a short discussion of future work.

2 Feature Extraction

2.1 Harris corner-detector

Image-feature detection is an important task in various vision applications. We use Harris corner-detector [6] to extract useful features. In the implementation, some heuristics can be applied to identifying image corners: (1) we can adjust the threshold value to limit the number of detected corners. (2) To prevent detecting too many corners within a small region, we can enforce a minimum distance between any pair of neighboring corners. In our implementation (targeting 352×240 resolution video sequences), we keep about $300 \sim 400$ corners in each video frame. The minimum distance is restricted to 5 pixels between any two corners. Moreover, sub-pixel precision techniques to further refine the corner location, usually achieved through a quadratic approximation, are not considered here due to concerns about the high computational cost.

The key advantage of a high-level, feature-based, background model over the pixel-level models is the substantial savings in computation cost for the processes of building and maintaining the model. For a 352×240 resolution video sequence, we need to process about 10^5 pixels in each frame, if a pixel-level background model is used. For our corner-based background model, however, only $300 \sim 400$

selected corners need be considered in the later steps. Furthermore, our optimized version of the Harris corner detector takes only a few milliseconds to detect corners in each video frame.

Because we use a set of sparse features, instead of all the pixels, to represent image and video sequences, an important question is whether, and how much, such a sampling would result in information loss and thus compromise the accuracy of detecting moving objects. Consider the following two scenarios of interest for a video surveillance scene:

- A moving object enters the homogenous sky or road surface. Originally, few corners would have occurred in such areas, but the intruder would instigate new corners, which indicate motion.
- A moving object enters the areas where a high density of corners has been detected over time. This novel object will introduce a number of corners that are likely to have different color and gradient properties from those of the background corners.

This simple analysis demonstrates the ability of a corner-based background model for effectively detecting moving-objects in video sequences. Next, we will introduce a SIFT-like descriptor to help with accurate identification and classification for each detected corner.

2.2 SIFT-like descriptor

Stable local-feature representation is a fundamental component of many image-understanding applications, such as object-recognition, image-matching, retrieval, etc. A local descriptor of a corner, ideally, should be distinctive (reliably distinguishing one corner of interest from others), precise, and robust with regard to small shifts and illumination changes. In [8], the authors compare and evaluate a number of descriptors, among which SIFT (Scale Invariant Feature Transform) [2] outperform others and prove to be highly robust to common image deformations. A SIFT local image descriptor is computed based on a histogram representation of image gradient orientations in its local neighborhood. More specifically, a 4×4 grid of histograms, each with eight orientation bins, effectively encodes the rough spatial structure of the image patch around the point of interest. The resulting 128-dimensional vector is then normalized to unit length.

The original objective of the SIFT-descriptor was developed for the image-matching task, where points corresponding to the same object are extracted from the images under different scales and views. Hence, the descriptors need to be scale-rotation-invariant. Our purpose is quite different. For background modelling, the corners, extracted from the same position in the background, operate under the same scale and rotation over time. Therefore, it is unnecessary to have a multi-scale implementation and orientation alignment, which are the major contributors to the high computational cost of building the original SIFT-descriptor. However, the following features of the standard SIFT-descriptor [2] are particularly useful for our application:

- A Gaussian weighting function is used to assign a weight to each pixel, in which the pixels farther away have less impact. The purpose of this Gaussian window is to provide robustness against boundary effects. That is, gradual changes in location will not result in sudden changes in the obtained feature vector.
- The descriptor could tolerate small localization errors through the creation of histograms over 4×4 sample sub-regions. Because the sub-pixel precision techniques, which could further refine the corner location, are not used in our system, it is inevitable that the positions of the detected corners might incur small shifts over time. This could cause unstable representations for the corners corresponding to the same background positions. Dividing the whole sample window into 4×4 sub-regions effectively alleviates such a negative effect.
- Linear illumination changes are implicitly eliminated since the descriptor only contains gradient orientation information. Non-linear illumination changes might result in large magnitudes for some gradients. Therefore, we limit the gradient magnitudes by applying a threshold to the unit feature vector (for example, limiting each bin value to 0.2). Then we re-normalize it to unit length. The value of 0.2 was suggested experimentally in [2].

In Figure 1, we make a direct comparison between two different types of local feature descriptors. We present two challenging scenes in the top row: the left image comes from a video clip containing swaying trees and significant illumination changes; the right image shows ripples on water. Red rectangles indicate the positions where a corner is supposed to be detected. We compare two types of corner descriptors: (1) SIFT-like Descriptor strictly follows the definition of SIFT but skipping a multi-scale implementation and orientation alignment, i.e. building gradient orientation histogram over 4×4 sample sub-regions and with post-steps of normalizing and thresholding the generated histogram. (2) $ASSET$-2 Descriptor exactly follows the definition in [10], i.e. the means of the smoothed image brightness and the **x, y** image derivatives. For the graphs on the 2^{nd} and the 3^{rd} rows, we plot the correlations of features detected and described over ten frames in each scene. Specially, the correlation is computed between a pair of local descriptors extracted from two consecutive frames, as

expressed in Equation (1).

$$Correlation = \frac{\sum_i U_i \times V_i}{\sqrt{\sum_i U_i^2 \times V_i^2}} \quad (1)$$

where U and V are two multi-dimensional feature vectors. Two identical vectors result in a maximum value of 1. For features with respect to the same object in the background, we expect a good descriptor to maintain a high correlation value over time, and thus we can detect a foreground object at this position once features with low correlation value have been observed. For both video clips, the SIFT-like descriptor significantly outperforms the $ASSET$-2 descriptor. This result clearly demonstrates that a few critical steps of the original SIFT definition, such as sampling over 4×4 sub-regions and some post-steps, significantly help with the stability of a local descriptor when small shifts and illumination changes occur.

The step of building SIFT-like descriptors demands significant computation; therefore we can afford to build descriptors for only a small number of corners. In our implementation, this step takes about 10 ms to generate the descriptors for all corners detected in a new video frame.

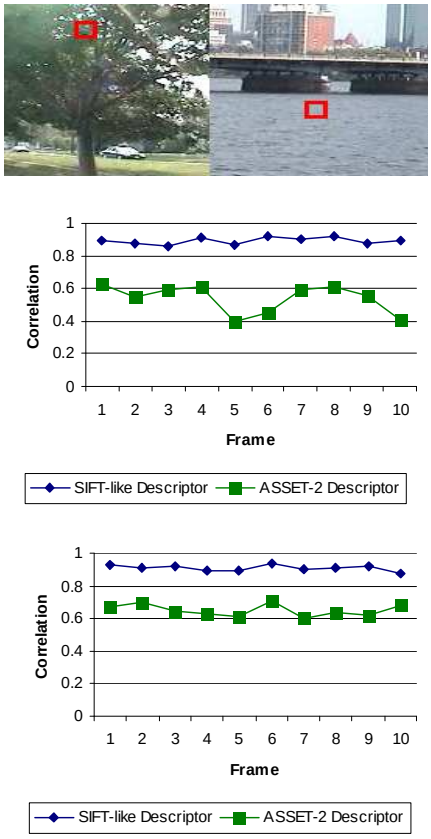

Figure 1. A comparison of descriptors

3 Background Modelling

Now with a set of detected image corners, each of which is represented by a 128-dimensional feature vector, the next step is to build and maintain a background model over time that in turn helps with effective detecting foreground moving-objects. In this section, we first introduce the data structure used in our corner-based background model. Then, we describe a dynamic process of learning this model over time. In the mean time, we use this dynamically learned model to classify each of the features detected in the current frame as either a background or a foreground feature.

3.1 Structure of model

In a pixel-based framework, each individual image can be treated as a two-dimensional matrix, where each matrix element records a history of the changes of the corresponding pixel. The exact information stored in this matrix, such as Gaussian parameters or properties of the predictive models, depends on the specific models used. In principle, a feature-based model differs from the pixel-based model in two ways. First, for the feature-based model, the image matrix is, in general, very sparse - most entries are empty. An entry is added only when a corner occurs in that position. Second, more information is maintained in each matrix entry and more complicated updating algorithm can be tolerated due to the significant saving in computation from the sparse representation of image. We represent each matrix entry as a 4-tuple vector as follows:

- Frequency of occurrence $\{frequency\}$. We increment the count if a corner is detected in the current frame. Moreover, this count automatically decays over time, thus gradually reducing the weights of records in past history.

- Mean of the descriptor $\{\overline{descriptor}\}$. For features that appear in the same location over time, we calculate the mean vector of the 128-dimensional descriptor. This mean vector compactly encodes the information of a particular location in the background, and later will be used to classify feature into either background or foreground.

- Mean and variance of correlation $\{\overline{correlation}, var\}$. We have already defined a vector correlation in Equation (1). This metric can be applied between the mean descriptor and the descriptor of newly detected features as well. We monitor the mean and the variance of this correlation over time. A low correlation indicates that the new features are probably from foreground objects. The variance is used to apply an adaptive threshold in feature classification step.

In the later discussion, this two-dimensional sparse matrix is referred to as model M. Each model entry $M(x; y)$ is a 4-tuple vector.

3.2 Dynamically learning

Based on a set of features detected and described in the previous steps, we aim to achieve two main goals during the learning step: background modelling and feature classification. We detail the learning and classification in Algorithm 1. In our algorithm, instead of blindly updating the background model, we choose the selective update scheme, i.e. only the background features classified in Step 3 are used to update model entry. Moreover, rather than using a direct mapping between position of model entry and feature location, a local window (e.g. 5×5) is defined to search the best model entry accounting for the newly detected feature. This shift matching design results in a very sparse representation of the learned model because an existing model entry will prohibit generating new entries within its neighborhood. Therefore, features subject to small shifts will be mapped onto the same model entry as time proceeds. In the experimental section, we will explain why this design is highly valuable for coping with some challenging dynamic scenes.

4 "Lucas-Kanade" tracker

A motion, caused by a real moving-object, should be highly spatial-temporal-correlated. In other words, a moving-object in a video sequence should be seen as the conjunction of several smoothed and coherent observations over time. In [7], a salient motion is defined as a motion that is likely to result from a typical surveillance target as opposed to other distracting motions (e.g., ocean waves and the oscillation of trees in the wind). They propose to detect the salient motions using the temporal integration of optical flow that is computed between two consecutive frames.

In our feature-based approach, a "$Lucas$-$Kanade$" feature tracker can be naturally integrated into the whole framework, which differs from the previous approaches in two ways:

- The tracker is directly applied to foreground features rather than every pixel or the whole frame. The foreground features, attached to a real moving-object in the image, should appear in the texture-rich regions, where the process of flow recovery is most well conditioned and where the information is most relevant. Therefore, our sparse representation of the image still keeps the most useful information while saving significant computation.
- The number of features identified as foreground is always much smaller than the number of the corners detected in each frame. And thus, we could design a powerful feature tracker without incurring high computational cost. Especially, our tracker acts as an independent "$agent$", which can deal with optic-flow calculation, merge into another tracker and delete itself.

Algorithm 1 *Modelling and Feature Classification*

***INPUT**: n video frames $I_1; :::; I_n$*
***OUTPUT**: features identified as foreground, $F_1; :::; F_m$*

For each new video frame do:

1. *Define a local 5×5 window around each detected feature in location (x, y).*

2. *In the local window, search for a model entry $M(x\prime, y\prime)$ whose mean descriptor has the maximum correlation with the feature under consideration.*

- *If $M(x\prime, y\prime)$ exists, jump to Step 3*
- *If return NULL, jump to Step 4*

3. *Maximum correlation obtained in Step 2 $> \overline{correlation}$ - 3 $\times$ var stored in $M(x\prime, y\prime)$*

- *True, a background feature, jump to Step 5*
- *False, a foreground feature, jump to Step 6*

4. *Allocate a 4-tuple vector, and attach to model M at position (x, y), jump to Step 6*

5. *Update each item of $M(x\prime, y\prime)$ based on the new feature.*

6. *For each entry in model M, decay its frequency. If it is reduced to 0, remove this entry from M.*

We detail the usage of a "$Lucas$-$Kanade$" tracker in Algorithm 2, which mainly consists of three individual modules. The first module takes charge of generating a new tracker for each newly identified foreground feature. The second module deals with the optic-flow calculation for each tracker in the list once a new frame becomes available. A number of rules are designed for tracker deletion and merger, which result in a significant reduction of the total tracker count. In addition, we pass the trackers with consistent trajectories to the third module, where a cluster of similar motion trajectories is confirmed as a real moving-object. The misclassified features, which are from the dynamic background, usually result in repetitive or random motion. Therefore, they can be removed in this step. In order to achieve sufficient tracking accuracy, we adopt an iterative implementation of the "$Lucas$-$Kanade$" optic-flow computation [1]. In our experiment, this step requires about $8 \sim 12$ ms per-frame.

Algorithm 2 *"$Lucas$-$Kanade$"Feature Tracker*

***INPUT**: m identified foreground features, $F_1; :::; F_m$*
***OUTPUT**: p confirmed moving-objects, $M_1; :::; M_p$*

For each newly identified foreground feature do (Module1):

- *Generate a "$Lucas$-$Kanade$" point tracker and add it to the tracker list.*
- *Each tracker records a trajectory of position changes $\{(X_0, Y_0), (X_1, Y_1)(X_i, Y_i)\}$ over time*

For each new video frame do (Module2):

- *Update each tracker's trajectory based on the new computed "$Lucas$-$Kanade$" optic-flow.*
- *Delete point trackers with inconsistent trajectory, e.g. small accumulated distance within 30 frames.*
- *For point trackers with consistent trajectory, e.g. accumulated distance within 30 frames exceeds 10 pixels,we add a new record to the motion-trajectory list and delete this tracker.*
- *Merge two trackers if across the same position (significantly reduces the total number of the tracked features).*

For all reported trajectories, do filtering (Module3):

- *Cluster similar trajectories to confirm a finding of real moving-objects.*
- *Eliminate noise trajectories*

5 Experiments and Discussions

Previously developed surveillance methods can effectively describe scenes that have a relatively static background and limited variations, but these are markedly less effective for handling dynamic backgrounds, such as the glinting of sunlight on water, the wafting of leaves in the wind, the movement of ocean waves, or variations in lights and shadows over a period of time. To demonstrate the strength of our proposed approach, we particularly tested and evaluated the ability of our corner-based background model on detecting and tracking moving-objects in such dynamic backgrounds.

5.1 A comparison to pixel-based model

In our experiments, we implemented a pixel-based background model, using Intel CVLib, to make a comparison with the proposed corner-based model. In Figure 2, we demonstrate a challenging surveillance scene, whose background provides significant illumination changes and contains waving trees which cover a significant area of the image. The first row shows four frames from a long video sequences. The images in the second row show the results using a pixel-based model. For the images in the third row, we show results of moving-object detection, using the proposed approach. Each circle represents a foreground feature confirmed by a "$Lucus$-$Kanade$" feature tracker, in which a consistent motion trajectory is observed in the subsequent frames. Clearly, the visualized result demonstrates that the proposed method achieves a very low false-positive and a high detection rate in identifying the true moving objects, whereas the pixel-based method seemed more easily confused by distracting motions in the dynamic backgrounds. Figure 3 presents another video example with very challenging situations. The small oscillation of the camera itself, along with the water ripples, produces significant distracting motions throughout the whole video sequence. Also, the size of the true moving object is in the distance thus seems relatively small and slow-moving. Superior and convincing results of the proposed method have been obtained for this video clip as well.

5.2 Dynamic scenes

Dynamic textures often exhibit repetitive patterns in the space-time domain. Therefore, a natural approach to model their behavior is via a "$pixel$-$process$" - that is, analyzing the values of an individual pixel over time. Our feature-based approach is superior to the traditional pixel-based methods in three aspects:

- Our SIFT-like descriptor achieves a stable representation of background features, whereas values of individual pixels are more vulnerable to image noise.
- Instead of exploiting the repetitive patterns of the temporal changes, our method is to explore and utilize the repetitive changes in the spatial domain. An important factor in the development of this new modelling technique is that, for a specific location in a dynamic background, features detected over time often encounter small shifts in the spatial domain. For example, a tree leaf waving in the wind may appear in a different location with a deviation of several pixels from the location observed in the previous/future frames. Therefore, we don't use a direct mapping between the location of model entries and the feature location during the learning of the model. A local window is defined to search for a model entry which best accounts for a newly detected feature. This shift matching design can effectively cope with many known dynamic scenes.
- A "$Lucas$-$Kanade$" tracker is used to monitor foreground features over time, where flow recovery is most well conditioned and where the information is most relevant. Because repetitive motions can result in inconsistent trajectories, and thus we can differentiate such dynamic motion-patterns from the interesting motions caused by foreground moving-objects.

5.3 Parameter setting and computation cost

In our featured-based approach, there are two tunable parameters. The first one is the maximum number of corners detected in each video frame. We do not need to specify this parameter in advance. Instead, it can be adaptively adjusted to the sensitivity of the system on-the-fly. For example, this number can be dynamically increased to enhance the sensitivity for detecting new intruders once a large number of foreground features with consistent trajectory have been observed. In the experiments, we found that, in most surveillance scenarios, $300 \sim 400$ corners are sufficient for keeping a high detection accuracy. Another important parameter is the size of the local window designed for the shift matching algorithm during background modelling. Usually, a large local window results in a very sparse learned model. The main advantage of a large local window is that background features subject to significant shifts will be mapped onto the same model entry as time proceeds. However, an overly sparse model will inevitably compromise the sensitivity of the system for detecting small moving-objects. Therefore, this parameter should be set at a value which strikes a good balance between these tradeoffs. In general, a relatively static background prefers a small local window size (e.g. 3×3), whereas the background presenting significant dynamic-motions demands a relatively large local window. As we are particularly interested in the dynamic scenes, a 5×5 local window is chosen for the testing videos in our experiments.

In addition to its effectiveness in modelling the dynamic scenes, our feature-based approach is highly efficient. Table 1 shows a comparison of the computation costs for a number of well-known methods. The Multiple Gaussian method [5] is a milestone work for background modelling, and the other two more recent methods are specifically designed to cope with the challenging dynamic scenes.

Methods	Frame size	Speed
Multiple Gaussian [5]	160×120	13 fps
Density Estimation [9]	160×120	7 fps
Autoregressive Model [12]	170×115	0.125 fps
Corner-based Model	352×240	25 fps

Table 1. A comparison of computation costs

6 Conclusions and future work

This paper has presented a sparse background model for detecting and tracking features over time, which is learned from a set of SIFT-like features and tries to exploit only the most useful parts (i.e. corners) of the image. Such a framework results in very significant savings in computation costs and effectively coping with dynamic scenes. We are investigating the use of other features, such as the edges and colors, to build the corner descriptor. In addition, we are designing a more comprehensive evaluation procedure for our method in order to make a thorough and logical comparison with other methods.

References

[1] J.-Y. Bouguet. Pyramidal implementation of the lucas kanade feature tracker description of the algorithm. *Intel Corporation*.

[2] D.G.Lowe. Object recognition from local scale-invariant features. *International Conference on Computer Vision*, September 1999.

[3] Elgammal, D. Harwood, and L. S. Davis. Non-parametric model for background subtraction. *European Conference on Computer Vision*, June 2000.

[4] N. Friedman and S. Russell. Image segmentation in video sequences: A probabilistic approach. *Thirteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, Augest 1997.

[5] W. Grimson, C. Stauffer, R. Romano, and L. Lee. Using adaptive tracking to classify and monitor activities in a site. *Computer Vision and Pattern Recognition*, June 1998.

[6] C. Harris and M. Stephens. A combined corner and edge detector. *Fourth Alvey Vision Conference*, pages 147–151, 1988.

[7] L.Wixson. Detecting salient motion by accumulating directionally-consistent flow. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(8):774–780.

[8] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *Computer Vision and Pattern Recognition*, June 2003.

[9] A. Mittal and N. Paragios. Motion-based background subtraction using adaptive kernel density estimation. *Computer Vision and Pattern Recognition*, June 2000.

[10] S. Smith. Asset-2: Real-time motion segmentation and shape tracking. *International Conference on Computer Vision*, October 1995.

[11] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7), July 1997.

[12] J. Zhong and S. Sclaroff. Segmenting foreground objects from a dynamic textured background via a robust kalman filter. *International Conference on Computer Vision*, October 2003.

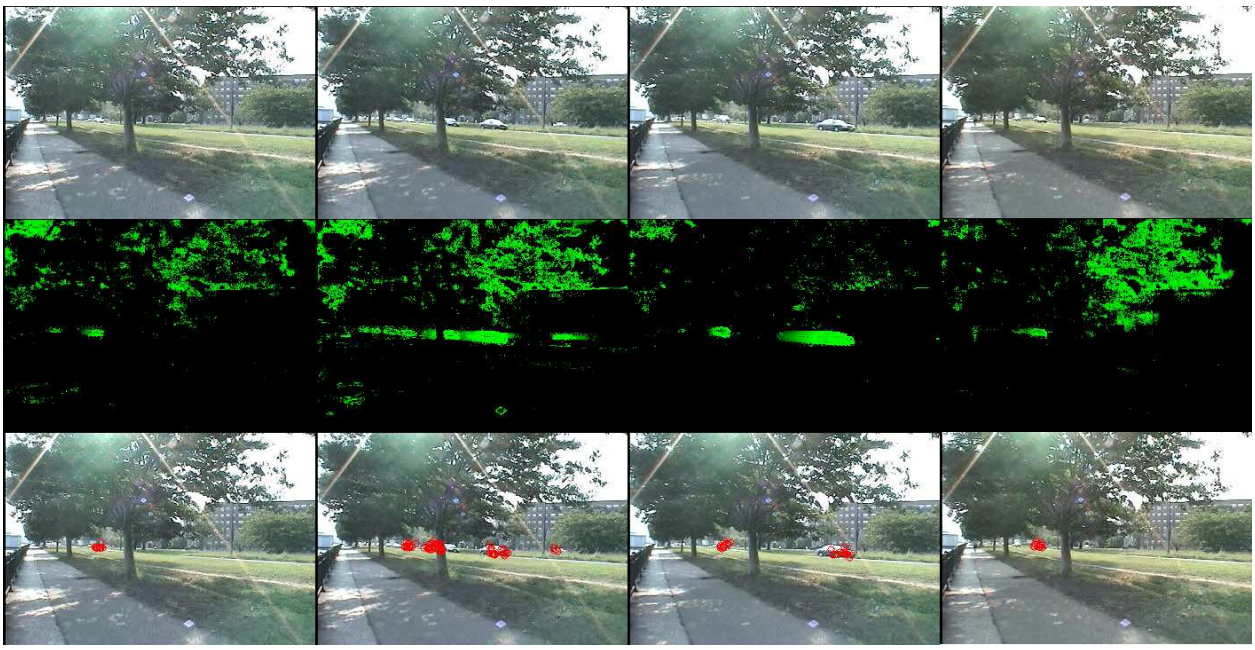

Figure 2. Video with moving-cars, trees swaying in the wind and significant illumination changes. *(7f, 467f, 770f, 1184f out of 1200 video frames in total)*

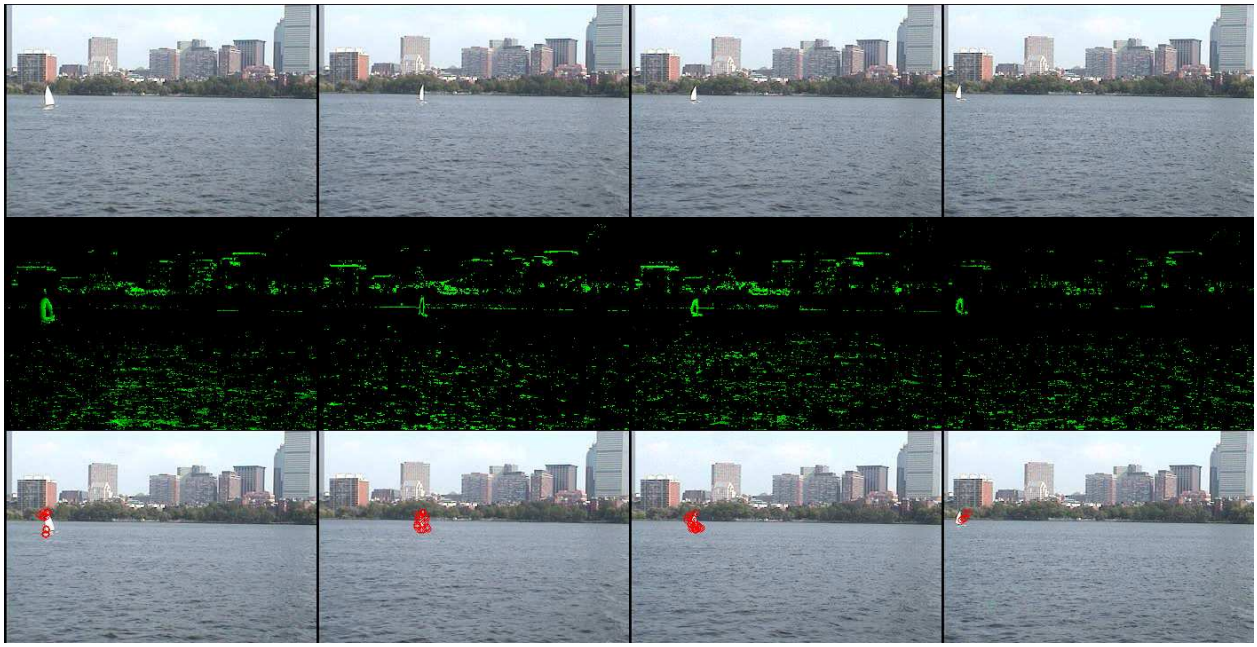

Figure 3. Video with a small boat and water ripples in the distance. *(21f, 569f, 970f, 1318f out of 1400 video frames in total)*