An Efficient Region-Based Background Subtraction Technique

Parisa Darvish Zadeh Varcheie, Michael Sills-Lavoie, and Guillaume-Alexandre Bilodeau
Department of Computer Engineering and Software Engineering
École Polytechnique de Montréal
P.O. Box 6079, Station Centre-ville Montréal (Québec), Canada, H3C 3A7
parisa.darvish-zadeh-varcheie@polymtl.ca, michael.sills-lavoie@polymtl.ca,
guillaume-alexandre.bilodeau@polymtl.ca

Abstract

This paper proposed an efficient region-based background subtraction technique using static camera for motion detection in various monitoring applications. Color histograms, texture information, and successive division of candidate rectangular image regions are utilized to model the background and detect the motion. The proposed method combines this principle and the Gaussian mixture background modeling to produce a new method which outperforms the classic Gaussian mixture background subtraction method. It has the advantages of filtering noise during image differentiation and providing a selectable level of detail for the contour of the moving shapes. Results show that our proposed method combined with Gaussian mixture outperforms results obtained with Gaussian mixture alone, particularly in the combination with MDPA distance.

1. Introduction

Background subtraction is a challenging computer vision problem. It is the basis of many algorithms and applications like video-surveillance to detect suspicious behaviours of people or objects. It is also applied in human-computer interactions to recognize the posture, gestures or activities of people.

The principle behind background subtraction is to subtract a background model image from the current frame. The resulting object motion consists of the differences between the two subtracted images. In practice, these differences do not always correspond to moving objects. They may correspond to shadows, changes of lighting or camera noise. Furthermore, some differences correspond to change in the image that researchers would like to ignore, like waving leafs and waves on a lake. The challenge is then to propose a background model that allows filtering these unavoidable perturbations, while still detecting correctly the moving objects of interest.

Many background detection methods were proposed with different background models and update strategies. Most of them rely on the difference between each pixel individually. Since perturbations often affect pixels individually, this cause misdetection when performing differentiation as observed by [1]. Our hypothesis is that using a neighbourhood around a pixel should allow filtering the perturbations affecting only few pixels in a region. We propose a method where background subtraction is performed iteratively from large to small rectangular regions using color histograms and a texture measure. Additionally, the classical Gaussian mixture method [2] is used on the smallest region level to improve even more the results. This is the contribution of this paper. We tested thoroughly our method by comparing the obtained moving regions with ground-truth regions using true and false positive rate measures, and we also characterized the impact of changes of parameters on the results to evaluate parameter sensibility and stability. Results show that our proposed method combined with Gaussian mixture outperforms results obtained with Gaussian mixture alone, particularly in the combination with MDPA distance [3].

One of the advantages of our proposed approach compared to the state-of-the-art is that it reduces the number of false detection as pixel-level differentiation may be performed only in region with significant motion. Another advantage is that the subdivision to go from large to smaller regions can be stopped before reaching the pixel level. Hence, if required, only a coarse background subtraction can be performed.

The paper is structured as follows. Section 2 gives an overview of the state-of-the-art for background subtraction methods. Section 3 presents our proposed background subtraction algorithm and section 4 demonstrates the ability of the proposed method with a thorough analysis. Finally, section 5 concludes the paper.

2. Related work

Among background subtraction methods, one of the most often used is the single Gaussian method [4]. In this method, statistics (mean and standard deviation) are computed for each background pixel. A pixel is considered as a foreground pixel, if its value when compared to its mean is larger than a threshold based on the standard deviation. This simple model does not cope well with perturbations in practice. Furthermore, it cannot handle well scenes with swaying vegetation or rippling water as it assumes a static background. Another method with a similar drawback is based only on the mean or the median of individual pixels [5]. In this work, the temporal average method is explored by testing different variations. Temporal average simply means to take the average RGB values for each background pixels over a certain number of frames and do a comparison with a threshold (not based on variance). The variations are on how the background is updated. They suggest different method to apply selective update for the pixels, so that only pixels corresponding to background are updated.

The simple Gaussian method can be improved significantly by using more than one Gaussian per pixel [2]. In this case, the *k* best distributions are selected dynamically, and a pixel is to be labelled as foreground if it is different from the *k* distributions based on standard deviation and mean. Both single Gaussian and Gaussian mixture models can handle illumination changes by updating dynamically the distributions. Many authors have proposed improvements to this algorithm. For example, to update the mixture model [6], or for adapting dynamically the number of distributions to consider for each pixel [7]. The pixel intensity distribution are not necessarily Gaussian, hence these models cannot always perform adequately.

Some region-based methods were proposed. Recently, a method based on local binary patterns was tested [1]. This method models each pixel using a binary pattern computed by comparing the value of neighbouring pixels with a center pixel. The binary pattern indicates for each neighbouring pixel if it is smaller or larger than the center pixel. Such binary patterns are calculated in a circular region around a given pixel, and the distribution of the binary patterns in the circular region and over time is used to model the pixel. Hence, foreground labelling decision are based on a region around a pixel. However, since a center pixel is used as reference for computing the binary pattern, a perturbation on this particular pixel can cause misdetection.

The methods presented in the work of Matsuyama *et al.* [9] and Mason and Duric [8] are related to our proposed method. In the work of Matsuyama *et al.* [9], change detection is achieved by correlation between blocks in the image with fixed blocks size. Because it uses correlation, this method may have difficulties with some dynamic scene. Using fixed block size makes it harder to balance robustness to noise and precision in the detection. In the work of Mason and Duric [8], fixed size block and histograms of edges are used. For the interested reader, other background subtraction algorithms are listed in [10].

3. Methodology

As all background subtraction approach, our proposed method is composed of a background model that is regularly updated and a measure of similarity to compare a given frame with the background model. The background is modeled at different scales with color histograms and texture of rectangular regions. The current frame in which we want to detect motion is modeled in the same way. Motion is detected by comparing the corresponding rectangular regions from the coarsest scale to the finest scale. Comparisons are done at a finer scale only if motion is detected at a coarser scale. Gaussian mixture background subtraction is used at finest scale. We call our method RECTGAUSS-Tex.

3.1 Modeling of rectangular regions

3.1.1 Background model and updating

Our background model M_R is simply based on statistics on the last RGB values of the pixels where no motion was detected. When we update the background, we substitute these pixels. We use substitution since color histograms account for pixels noise.

The background model M_R is built using pixel statistics of rectangular regions. The background image is first divided into 4 by 3 pixels rectangles. We use this size of rectangle because it corresponds to the aspect ratio of 4:3 images, and because images are described at different scales using a rectangle hierarchy, where four rectangles of size 4 by 3 give rise at a coarser scale to a rectangle of size 8 by 6, and so on. For each 4 by 3 rectangle (the finest scale), we compute two statistical measures. These measures will allow us to detect changes between M_R and the current image I_R. The first measure is the classical color histogram H_M and the second measure is the variance V_M of the pixels in the region. This second measure is to model the texture in a rectangle. It is similar to the method used in the work of Shen *et al.* [11]. For the color histograms, we use a uniform K bins quantification (usually 64) for each RGB channel. The variance is computed from the intensity of pixels in the region. After these computations we move to a coarser scale. We regroup four 4 by 3 rectangles to construct 8 by 6 pixels rectangles. Each rectangle at the 4 by 3 level is used only once to create a 8 by 6 rectangle. The statistical measures at this new scale are computed by merging the statistics of the 4 by 3 rectangles. Grouping of

rectangles and statistics merging are performed until a user defined number of coarse rectangles R_c is obtained (usually 100).

3.1.2 Motion detection

To detect motion, the current frame I_R is modeled in the same way as the background by computing statistics on a set of rectangles at different scales. Motion detection is illustrated by Fig. 1. Differences between I_R and M_R are computed as follows.

Starting from the rectangles at the coarsest scale and for each rectangle, we compute the histogram similarity using the MDPA distance [3] divided by the number of pixels in the rectangle. That is, the similarity S_H is

$$S_H(H_I(i,j), H_M(i,j)) = \frac{\sum_{b=0}^{K-1} \left| \sum_{k=0}^{b} (H_I(i,j)[k] - H_M(i,j)[k]) \right|}{\sum_{b=0}^{K-1} H_I(i,j)[b]}, \quad (1)$$

where $H_I(i,j)$ and $H_M(i,j)$ denote histograms of corresponding rectangles in the image and in the model, and k is the k^{th} bin of the K bin histograms. It is also possible to compare the histograms according to the Euclidean distance [3], but the results are not as good as MDPA (section 4). S_H is:

$$S_H(H_I(i,j), H_M(i,j)) = \sqrt{\sum_{b=0}^{K-1} (H_I(i,j)[b] - H_M(i,j)[b])^2} \quad (2)$$

Two histograms are similar if

$$S_H(H_I(i,j), H_M(i,j)) < T, \quad (3)$$

where T is a similarity threshold.

As illustrated by Fig. 1, if two histograms are similar, we simply copy the rectangle of I_R into the background image to update it. In Fig. 1, black rectangles correspond to rectangles copied to the background model, the moving object being the person. Otherwise, if the histograms are different, we move to the next finer scale to locate motion more precisely by exploiting smaller rectangles. To make sure to detect moving regions located near the rectangle for which we just established having different histograms, as we move to the finer scale, we also check neighbouring rectangles (see Fig. 1). Indeed, neighbouring rectangles might contain small regions connected to larger one in other rectangles. These small regions might not cause a change important enough in the histogram of the previous scale.

At all scales, the same computations are performed until the finest scale is reached. However, threshold T is modified at each scale to account for a fewest number of pixels. To avoid the influence of noisy pixels, as rectangles get smaller the threshold is increase by parameter ΔT (i.e. $T = T + \Delta T$).

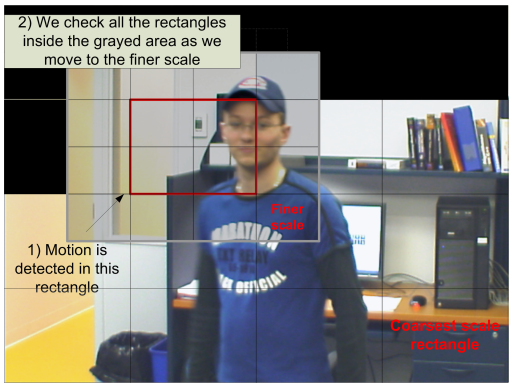


Figure 1. Illustration of motion detection as the algorithm progresses into scale. Black rectangles illustrate regions labeled as background. Motion is detected in the indicated rectangle. When, the algorithm moves to a finer scale, it will consider all the rectangle in the grayed area.

For more robustness, a second statistical measure is used. This second measure is the variance V_M of the pixels in the region. This second measure is to model the texture in a rectangle. The variance is computed from the intensity of pixels in the region. Using the measurement of Shen *et al.* [11]. The variance of the intensity of pixels is also computed for each rectangle at all scales.

The texture information is used to modulate threshold T based on the probability of finding a moving object. Indeed, it was found that this simple texture measure was more useful to modulate histogram similarly thresholding then when used by itself as a second criterion for foreground detection. Hence, texture is used to module detection sensibility for case where textures in the rectangle areas are different, providing a cue about a higher probability of motion.

Given the variance of a rectangle from the background model V_M and the variance of a rectangle from the current image V_I, a rectangle is said to be textured if its variance is larger than the threshold T_v. If corresponding rectangles of I_R and M_R are both not textured, than most probably no changes happened. So we can be more severe for the histogram comparison. In this case, threshold T is increase by 50% with T=$T * 1.5$. If one rectangle of I_R or M_R is textured, than it is assumed that a change happened and the threshold is decreased with $T = T * 0.8$ (less severe). Finally, if both rectangles are textured the following equation is applied to verify texture similarity:

$$TS = \frac{\left|\overline{Vec_M^T} \cdot \overline{Vec_I}\right|}{\left\|\overline{Vec_M}\right\| \cdot \left\|\overline{Vec_I}\right\|}. \quad (4)$$

This gives as a result a value between 0 and 1 corresponding to the similarity of the intensities of the rectangle (in vector format) in the current image Vec_I and the background model Vec_M. A result of 1 means that both rectangles have exactly the same texture. Threshold T is adjusted by $T = T * (0.8 + TS)$.

The equations to adjust T were determined based on experiments.

3.2 Combination with Gaussian mixture

Thru testing, we have noticed that our method is improved by combining it with Gaussian mixture [**?**] to obtain accurate regions at the pixel level. Indeed, the precision of regions obtained with our method is limited by the smallest rectangle size which is 4 by 3 pixels. Furthermore, Gaussian mixture is improved as a preliminary segmentation is already performed allowing it to be more permissive.

3.2.1 Gaussion mixture method

In the Gaussian mixture method, each position in the image is modeled as a pixel process with a Gaussian distribution

$$\eta(X_t, \mu_t, \Sigma_t) = \frac{1}{(2\pi)^{n/2}\left|\Sigma_t\right|^{1/2}} e^{-\frac{1}{2}(X_t-\mu_t)^T \Sigma_t^{-1}(X_t-\mu_t)}, \quad (5)$$

where X_t is a measure, μ_t is the mean and Σ_t is the covariance of the distribution. To account for the fact that the background might need to be represented by more than one distribution, for each pixel, the background model is represented by K distributions. The probability that a measure X_t belong to the background is

$$P(X_t) = \sum_{i=1}^{K} \omega_{i,t} * \eta(X_t, \mu_{i,t}, \Sigma_{i,t}), \quad (6)$$

where $\omega_{i,t}$ is the weight of the i^{th} distribution. In practice, we assume that each RGB channel are independent and of the same variance. Hence, a pixel position is modeled by K Gaussian distributions with a given mean $\mu_t = [\mu_{t,r}, \mu_{t,g}, \mu_{t,b}]$ (one mean for each channel) and variance σ_t^2. A pixel X_t is a moving pixel if

$$|X_t - \mu_t| > T_g \sigma_t, \quad (7)$$

for the B best distributions, where T_g is a threshold to be fixed. Two additionnal parameters are the updating rate α for the distributions and T_b to choose the B best distributions among the K distributions.

3.2.2 Gaussion mixture in our method

In our method, the Gaussian mixture method will be applied when the finest scale is reached. This means that the Gaussian mixture background model is updated with the rectangles where no motion is detected. For rectangles where motion is detected at the finest scale (3 by 4 pixels), Gaussian mixture motion detection is applied on the rectangle. Hence, Gaussian mixture motion detection is applied only for regions where there is obvious motion.

4 Experiments and analysis

We have characterized thoroughly our method with various experiments. The goal of this work is to propose a background subtraction method that is efficient, but that does not need too much parameters tuning to be easily applicable. Hence the performance of RECTGAUSS-Tex and RECTGAUSS-Tex-Euc (which is the proposed method using Euclidean distance) were compared with available background subtraction methods. The actual foreground/background segmentations were verified with ground-truth and with the same parameters for the group of videos in each dataset.

4.1 How we tested

4.1.1 Data and ground truth

We have used the dataset of Wallflower [12] to test our method in different situations (Bootstrap, Camouflage, Foreground aperture, Light switch, Moved object, and Time of day). We have also used the ground-truth accompanying this dataset. However, this dataset dates from a couple of years, and the resolution is very low (160x120). So we made a couple of homemade videos to test the algorithm with higher resolutions. *IpCapAutoEx* is a 320x240 video sequence where an IP security camera is set in auto-exposure mode. As a person walks by, the colors of pixels change as the camera adapts to the scene content. *BagPick-up* is a 640x480 video captured with a machine vision camera where a person picks-up a bag already present in the scene. Furthermore, the motion of the walker causes a curtain to move. *Walker* is a 640x480 video captured with an IP security camera where a walker enters a room, turns around, turn off the light and then exits. *AtriumLassonde* is a 640x480 video where many walkers adopt random trajectories. The capture is particularly noisy. We created about 6 ground-truths per video (total of 23) for these four videos.

4.1.2 Description of experiments

For both datasets different background subtraction methods are tested. These techniques are Simple Gaussian (SG) [4]

Table 1. Parameters for the experiment using wallflower dataset.

Method	K	T_g	α	T_b	T	ΔT	R_c	T_v	E
SG	-	-	0.05	-	-	-	-	-	-
GM	3	3.5	0.007	0.5	-	-	-	-	-
TA	-	-	-	-	-	-	-	-	31
RGT-Euc	5	3	0.005	0.3	0.047	0.02	100	2	-
RGT	5	3	0.005	0.3	0.044	0.014	100	2	-

K: Number of Gaussian distributions, T_g: foreground/background threshold (in number of standard deviation), α: Learning rate, T_b: Proportion of background distribution, T: Rectangle similarity threshold, ΔT:Rectangle similarity threshold increase, R_c: Number of rectangles, T_v: Rectangle texture similarity threshold, E: Similarity threshold (in RGB levels).

that is implemented by OpenCV, Gaussian Mixture (GM) [2], Temporal Average (TA) [5] and the proposed method which is called RECTGAUSS-Tex(RGT) approach. Also the proposed technique is compared in the case of using Euclidean distance instead of MDPA (RGT-Euc). For each background subtraction technique, there are some sets of parameters that should be determined. A suitable set of parameters for a set of video files is the one that entails the most correctly identified pixels for all of the video files in the set. Here, we swept the parameter space to obtain the appropriate set of parameters. Since we use two different datasets (Wallflower and our video files), we repeated this procedure to obtain two independent set of parameters. They are given in Table 1 and Table 2.

For the performance evaluation of our proposed background subtraction technique, two metrics are used. These metrics are True Positive Rate (TPR) and False Positive Rate (FPR). True Positive (TP)/True Negative (TN) are defined as the number of foreground/background pixels that are correctly classified as foreground/background pixels. False Positive (FP)/False Negative (FN) are defined as the number of background/foreground pixels that are erroneously classified as foreground/background. The TPR and FPR are defined as

$$TPR = \frac{TP}{TP + FN} \tag{8}$$

and

$$FPR = \frac{FP}{FP + TN}. \tag{9}$$

If $TP \gg FN$, the TPR will be high. High TPR will be obtained if the number of real foreground pixels that are detected in the extracted foreground is much larger than the number of real foreground pixels that are detected in the background. Indeed, less parts of the real foreground are lost. The TPR is not the only criterion for the evaluation of a background subtraction technique. It is just about the real foreground pixels in which some of them will be detected as foreground called TP and the others will be detected as background called FN. It is also important to investigate the influence of the presence of real background pixels in the extracted foreground. This can be computed with the false positive rate (FPR). For selecting the best technique, it is important to have a high True Negative Rate (TNR) or a low FPR (i.e. TNR=1-FPR). If $TN \gg FP$, the FPR value will be small. Low FPR (or high TNR) means classifying most parts of the background as background. The technique that has the highest TPR and the lowest FPR will be the best solution for background subtraction.

Table 2. Parameters for the experiment using our dataset.

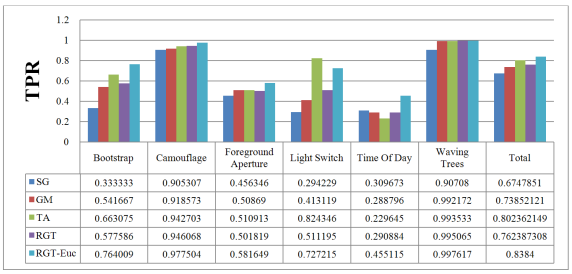
Method	K	T_g	α	T_b	T	ΔT	R_c	T_v	E
GM	3	3.4	0.009	0.5	-	-	-	-	-
SG	-	-	0.05	-	-	-	-	-	-
TA	-	-	-	-	-	-	-	-	31
RGT-Euc	5	3	0.01	0.3	0.3	0.1	100	2	-
RGT	5	3	0.01	0.3	0.38	0.0095	100	2	-

K: Number of Gaussian distributions, T_g: foreground/background threshold (in number of standard deviation), α: Learning rate, T_b: Proportion of background distribution, T: Rectangle similarity threshold, ΔT:Rectangle similarity threshold increase, R_c: Number of rectangles, T_v: Rectangle texture similarity threshold.

4.2 Results and discussions

Fig. 2 and Fig. 3 show the TPR and FPR values for four different background subtraction techniques for the Wallflower dataset. The optimum set of parameters used for each technique is listed in Table 1. The best technique should have highest TPR value. Simple Gaussian method that is implemented in OpenCV library has the lowest TPR in general for all the videos. According to these results, Temporal Average and RECTGAUSS-Tex-Euc have the highest TPR. TPR values of Gaussian mixture method which is commonly used in computer vision tasks are often smaller than these two techniques.

But for complete separation of foreground from background, the FPR value has to be very small (i.e. ideally equal to zero). Indeed, if the FPR value is not small, it means that most parts of the image are detected as foreground. In this case, the background subtraction technique is not appropriate because it does not label background pixels correctly.

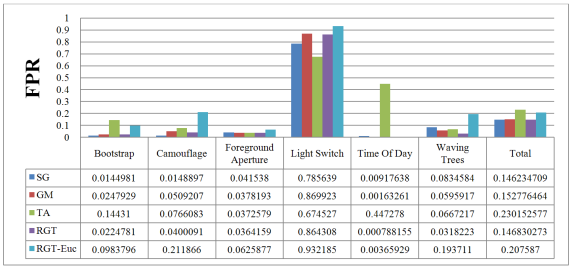


	Bootstrap	Camouflage	Foreground Aperture	Light Switch	Time Of Day	Waving Trees	Total
SG	0.333333	0.905307	0.456346	0.294229	0.309673	0.90708	0.6747851
GM	0.541667	0.918573	0.50869	0.413119	0.288796	0.992172	0.73852121
TA	0.663075	0.942703	0.510913	0.824346	0.229645	0.993533	0.802362149
RGT	0.577586	0.946068	0.501819	0.511195	0.290884	0.995065	0.762387308
RGT-Euc	0.764009	0.977504	0.581649	0.727215	0.455115	0.997617	0.8384

Figure 2. True Positive Rate of different background subtraction techniques for Wallflower dataset. The Total column is for all the videos combined together.

Fig. 3 shows the result for the FPR. Temporal Average has the highest FPR. This means that it is not really a good method. The FPR of our technique with MDPA is among the smallest in general among all the videos. This means that our method combines a high TPR and a small FPR. The behaviour of Gaussian mixture method is similar, but it is outperformed by our method.

According to the results using MDPA is much more ef-



	Bootstrap	Camouflage	Foreground Aperture	Light Switch	Time Of Day	Waving Trees	Total
SG	0.0144981	0.0148897	0.041538	0.785639	0.00917638	0.0834584	0.146234709
GM	0.0247929	0.0509207	0.0378193	0.869923	0.00163261	0.0595917	0.152776464
TA	0.14431	0.0766083	0.0372579	0.674527	0.447278	0.0667217	0.230152577
RGT	0.0224781	0.0400091	0.0364159	0.864308	0.000788155	0.0318223	0.146830273
RGT-Euc	0.0983796	0.211866	0.0625877	0.932185	0.00365929	0.193711	0.207587

Figure 3. False Positive Rate of different background subtraction techniques for Wallflower dataset. The Total column is for all the videos combined together.

ficient than using Euclidean distance. Indeed the MDPA is more precise than the Euclidean distance. There is no motion for the *Moved object* video file of the Wallflower dataset. Thus TPR and FPR values of all background subtraction techniques are equal to zero. This is why this result is not shown in the figures. Fig. 4 shows results of the proposed background subtraction for a frame of Wallflower dataset. For comparing the proposed method, the ideal results of background subtraction and also proposed algorithm using Euclidean distance are given. There are more false positive parts in the case of Euclidean distance as shown in $Bootstrap$, $WavingTrees$, $Camouflage$ and $ForegroundAperture$ videos.

According to the TPR and FPR values, all the techniques

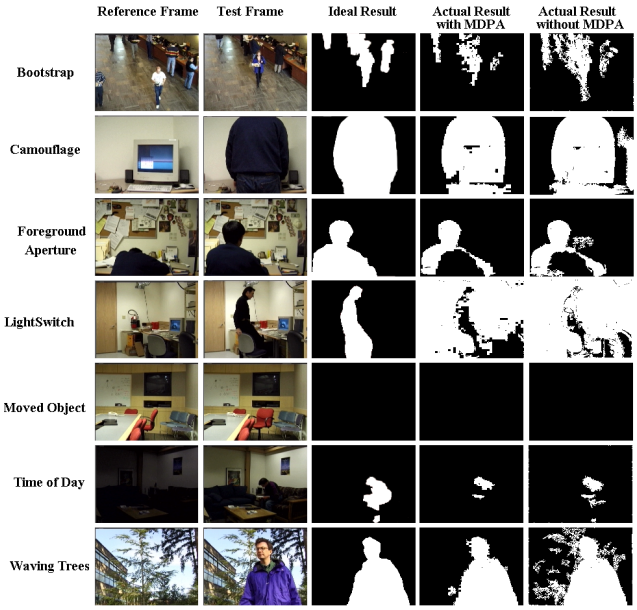
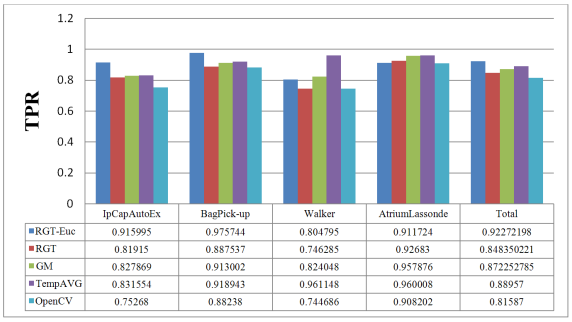
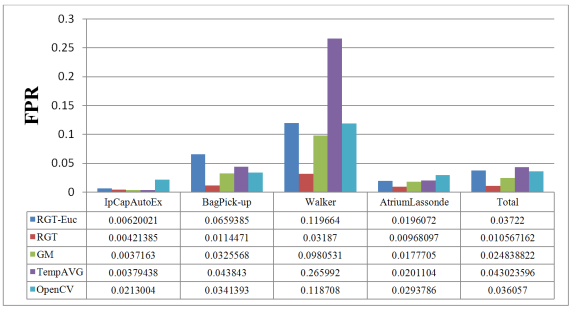


Figure 4. Detection results of the proposed method for Wallflower dataset.



	IpCapAutoEx	BagPick-up	Walker	AtriumLassonde	Total
RGT-Euc	0.915995	0.975744	0.804795	0.911724	0.92272198
RGT	0.81915	0.887537	0.746285	0.92683	0.848350221
GM	0.827869	0.913002	0.824048	0.957876	0.872252785
TempAVG	0.831554	0.918943	0.961148	0.960008	0.88957
OpenCV	0.75268	0.88238	0.744686	0.908202	0.81587

Figure 5. True Positive Rate of different variations of proposed background subtraction technique for our dataset. The Total column is for all the videos combined together.



	IpCapAutoEx	BagPick-up	Walker	AtriumLassonde	Total
RGT-Euc	0.00620021	0.0659385	0.119664	0.0196072	0.03722
RGT	0.00421385	0.0114471	0.03187	0.00968097	0.010567162
GM	0.0037163	0.0325568	0.0980531	0.0177705	0.024838822
TempAVG	0.00379438	0.043843	0.265992	0.0201104	0.043023596
OpenCV	0.0213004	0.0341393	0.118708	0.0293786	0.036057

Figure 6. False Positive Rate of different variations of proposed background subtraction technique for our dataset. The Total column is for all the videos combined together.

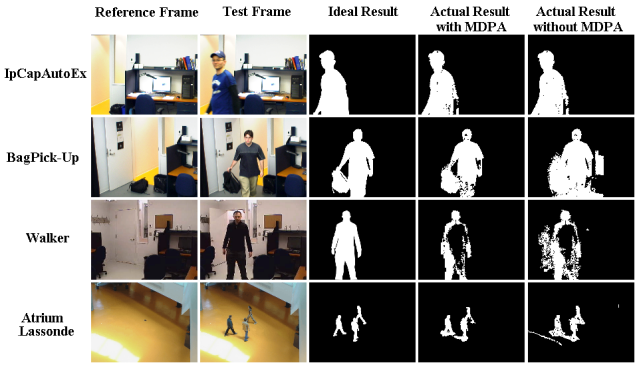


Figure 7. Detection results of the proposed method for our dataset.

are very sensitive to the sudden change of illumination. For this event, Temporal Average method has the highest TPR and smallest FPR values.

In the second experiment, we tested our technique with our dataset that contains different video files. The optimum set of parameters of each technique is in Table 2. Fig. 5 and 6 give the TPR and FPR values for each video. There is also a light switch for $Walker$ video file. It makes some higher FPR in the results. In addition, we computed the total TPR and FPR values of these four video files together. It is similar to have a video file that is the combination of different categories of video files. The total TPR and FPR values are calculated to have a general evaluation.

Rectangle size depends on the size of the object to detect. Fewer rectangles mean larger ones at the coarsest level, and hence more filtering of small motion. In the homemade videos used, the moving objects are relatively large. This is why 100 rectangles is enough at the coarsest level. According to these figures, RECTGAUSS-Tex-Euc, Temporal Average and Gaussian Mixture have the highest amount of TPR respectively. But as discussed before, the FPR values of these three techniques are high. The RECTGAUSS-Tex has the smallest FPR. In addition the TPR value of this technique is very close to the highest TPR for these three background subtraction methods. Hence, utilizing the combination of both texture property and Gaussian mixture method on the rectangular region modeling makes RECTGAUSS-tex an improvement over Gaussian mixture, because it has about higher TPR, but a smaller FPR. It means that fewer errors are included in the foreground. In addition applying the MDPA distance entails less FPR in the detected foreground. Fig. 7 shows the detection results of proposed method for our dataset. For comparison, similarly to Fig. 4 the ideal results and proposed method using Euclidean distance results are given. Here again by using Euclidean distance more false positive parts are detected as illustrated for $BagPick-Up$, $Walker$ and $AtriumLassonde$ videos.

Table 3. Processing rate (fps) of different background subtraction for different image resolutions.

Resolution	Processing time of GM (fps)	Processing time of RGT-Euc (fps)	Processing time of RGT (fps)
640 x 480	7.33	5.53	4.56
320 x 240	28.3	22.39	16.1
160 x 120	86.4	49.5	48

As for the performance of the algorithm, Table 3 shows the comparison of the execution time of Gaussian Mixture method and proposed background subtraction method with and without using MDPA. In addition the effect of different resolutions in the execution time is shown. These algorithms were implemented on a 2.66 GHz Xeon(R) in C++ using OpenCV. In the comparison of the execution time of the proposed algorithm with Gaussian Mixture, our algorithm has an increase of 160% for video files with 640 x 480 resolutions, 175% for video files with 320 x 240 resolutions and 179% for video files with 160 x 120 resolutions for total number of frames. By combining the MDPA distance the execution time is increased additionally by 121% for video files with 640 x 480 resolutions, 138% for video file with 320x240 resolutions and 102% for video file with 160 x 120 resolutions for total number of frames. For the proposed algorithm we obtained 4.56 fps for the processing of 807 frames with 640 x 480 image resolutions (our dataset), 16.1 fps for 1500 frames with 320 x 240 image resolutions and 48 fps for 16158 frames with 160 x 120 image resolutions. So this algorithm can be used for real-time applications with image resolution of 320 x 240 or less. The performance hit is mostly caused by the MDPA distance to compare histogram that is much slower than the Euclidian

distance. We selected MDPA because it is a more accurate distance measure as it accounts for the error distribution among the histogram bins. Indeed MDPA distance calculate the histograms differences adaptively for each bin. By using Euclidian distance, we get faster processing rate, but the results are not as good as with MDPA.

5 Conclusion

In this paper, a novel background subtraction technique is proposed. The background is modeled by rectangular regions described by a color histogram and a texture measure. It is model at different scales to detect motion more and more precisely. This background model is combined with the Gaussian mixture model. The use of rectangular regions filters out small motion like swaying vegetations or waves, and noise coming from data acquisition. The Gaussian mixture background subtraction then completes the work by detailing the foreground detection in rectangular areas where significant motion is detected.

The algorithm was evaluated with various video files against different change of illuminations and resolutions. The results obtained show that RECTGAUSS-Tex outperforms Gaussian mixture as it has a similar TPR, but a smaller FPR.

The drawback of the method is that it takes more processing time than Gaussian mixture. This technique is appropriate for real-time application with image resolution of 320 x 240 or less. However, MDPA distance calculations could be performed in parallel to speed-up processing. Future work is to adjust the thresholds dynamically based on scene contents and object appearance. Hence, if an object has similar color to the background, the detection threshold could be decrease in this background area for more sensitivity.

Acknowledgment

This work is supported by the Fonds Québécois de la Recherche sur la Nature et les Technologies (FQRNT), by the Natural Sciences and Engineering Research Council of Canada (NSERC), and by Canada Foundation for innovation (CFI).

References

[1] M. Heikkila, M. Pietikainen, "A Texture-Based Method for Modeling the Background and Detecting Moving Objects," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 28, no. 4, pp. 657-662, 2006.

[2] C. Stauffer, W. E. L. Grimson, "Adaptive Background Mixture Models for Real-Time Tracking," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition*, vol. 2, pp. 246-252, 2000.

[3] S. H. Cha, S. N. Srihari, "On measuring the distance between histograms," *Pattern Recognition*, vol. 35, no. 6, pp. 1355-1370, 2002.

[4] S. J. McKenna, S. Jabri, Z. Duric, A. Rosenfeld, H. Wechsler, "Tracking Groups of People," *Computer Vision and Image Understanding*, vol. 80, no.1, pp. 42-56, 2000.

[5] B. Shoushtarian, H. E. Bez, "A practical adaptive approach for dynamic background subtraction using an invariant colour model and object tracking," *Pattern Recognition Letters*, vol. 26, no. 1, pp. 5-26, 2005.

[6] P. KaewTraKulPong, R. Bowden, "An Improved Adaptive Background Mixture Model for Real-Time Tracking with Shadow Detection," *Proc. 2nd European Workshop on Advanced Video Based Surveillance Systems*, *AVBS*01, Kingston, UK, 2001.

[7] Z. Zivkovic, "Improved Adaptive Gaussian Mixture Model for Background Subtraction," *Proc. International Conf. Pattern Recognition*, vol. 2, pp. 28-31, 2004.

[8] M. Mason, Z. Duric, "Using Histograms to Detect and Track Objects in Color Video," *Proc. Applied Imagery Pattern Recognition Workshop*, pp. 154-159, 2001.

[9] T. Matsuyama, T. Ohya, H. Habe, "Background Subtraction for Non-Stationary Scenes," *Proc. Asian Conference Computer Vision*, pp. 622-667, 2000.

[10] R. Cucchiara, C. Grana, M. Piccardi, A. Prati, "Detecting Moving Objects, Ghosts, and Shadows in Video Streams," *IEEE, Transaction Pattern Analysis Machine Intelligence*, vol. 25, no.10, pp. 1337-1342, 2003.

[11] C. Shen, X. Lin, Y. Shi, "Moving object tracking under varying illumination conditions," *Pattern Recognition Letters*, vol. 27, no. 14, pp. 1632-1643, 2006.

[12] K. Toyama, J. Krumm, B. Brumitt, B. Meyers, "Wallflower: principles and practice of background maintenance," *Proc. Seventh IEEE, International Conference on Computer Vision*, pp. 255-261, 2002.