Practical Mixtures of Gaussians with Brightness Monitoring

Stefan Atev, Osama Masoud, Nikos Papanikolopoulos
Dept. of Computer Science and Engineering
University of Minnesota

Abstract— **We discuss some of the practical issues concerning the use of mixtures of Gaussians for background segmentation in outdoor scenes, including the choice of parameters. Different covariance representations and their performance impact are examined. In addition, we propose a simple, yet efficient method for coping with sudden global illumination changes based on smoothing brightness and contrast changes over time. All of the discussed methods are capable of running in real time at reasonable resolution on current generation PCs.**

Keywords: Background segmentation, illumination monitoring, mixture models.

I. INTRODUCTION

Robust tracking in outdoor scenes is an important problem in surveillance, data collection and monitoring applications. A common approach to the problem is to segment the frames of an input video sequence into background and foreground regions, and then use the regions as primitive features.

Stauffer et al. [3] proposed a widely used method for background segmentation using a mixture of Gaussians to model the processes generating each pixel in a video sequence. The algorithm is appealing because it is adaptive, online, and allows for multimodal background processes.

We discuss some problems with applying the method that arise in a particular traffic intersection monitoring application [1]. Difficulties encountered ranged from choosing suitable parameters for the method to issues caused by the camera's automatic exposure circuitry. Since the method dominates the running time of our monitoring application, various optimizations were used in the implementation.

The rapid increase in the processing power of ordinary PCs also allowed us to experiment with modifications of the algorithm suggested by the authors, but rejected for performance reasons. We suggest that some of the performance/quality tradeoffs in the original postulation of the algorithm can be re-evaluated.

II. RELATED WORK

The mixture of Gaussians approach to background segmentation was successfully used in [3] in a long-term traffic surveillance application. The version of the algorithm used in [6] for both traffic monitoring and video meeting applications allowed for detection of regions in the image where the background was never observed (which proved useful for video meeting applications). A thorough theoretical treatment of the mixture of Gaussians approach is presented in [4], where improvements over the base algorithm such as hysteretic Gaussian match threshold are suggested. The paper [4] also discusses some post-processing of the background mask and suggest that a illumination monitor such as the one described here may be necessary for practical use of the method. The focus of [5] is on the issue of initializing mixture models in order to alleviate false initialization problems and to decrease the amount of processing necessary to obtain accurate classification. We have had good success in post-processing the segmentation results using the algorithm presented in [7].

III. OVERVIEW OF THE ALGORITHM

It was proposed in [3] that each pixel in the input frame be modeled as a weighted combination of several normal distributions. We present the method briefly, but omit some time-indicating indices in our notation for conciseness.

The probability of observing a given pixel value $c \in R^p$ at some time instant is given by

$$P(c) = \sum_{i=1}^{n} P(M_i)P(c|M_i) \qquad (1)$$

Here M_i is the i-th mixture component, represented by a mean μ_i and a $p \times p$ covariance matrix Σ_i. $P(M_i)$ is the mixing coefficient corresponding to the i-th mixture component and $P(c|M_i)$ is c's likelihood according to that mixture component. The individual mixture components are Gaussians, so we have:

$$P(c|M_i) = N_p(c, \mu_i, \Sigma_i) = \frac{e^{-\frac{1}{2}(c-\mu_i)^T \Sigma_i^{-1}(c-\mu_i)}}{\sqrt{(2\pi)^p|\Sigma|}} \qquad (2)$$

The mixture components are ordered so that an observed pixel value c is matched against more probable components first. Based on the observation that the noise in a pixel process is usually proportional to its intensity, Stauffer et al. proposed that the mixture components be ordered by their $P(M_i)/\sqrt{|\Sigma_i|}$ ratio.

A pixel c is said to match M_i if c is within δ[1] standard deviations of μ_i. In other words, c matches M_i if

$$\sqrt{(c-\mu_i)^T \Sigma_i^{-1}(c-\mu_i)} < \delta \qquad (3)$$

[1]The choice of δ has little impact on the performance of the algorithm and is set to 2.5 in [3], [4]

Once it has been determined that c matches a mixture component M_k, it must be determined whether M_k is a part of the background process or not. Given a threshold $B \in [0..1]$, M_k is considered a part of the background if $\sum_{i=1}^{k-1} P(M_k) \leq B$.

All unmatched mixture components retain their mean and covariance unchanged, and their likelihood is updated as $P(M_i) \leftarrow (1-\alpha)P(M_i)$. If no component matches the pixel, the last (i.e. least probable) mixture component is replaced by a new distribution with initial mean equal to c, a small mixing component and a high variance. If a distribution M_k is matched, we use the following update equations (applied in the given order):

$$P(M_k) \leftarrow (1-\alpha)P(M_k)+\alpha \quad (4)$$

$$\mu_k \leftarrow (1-\beta)\mu_k+\beta c \quad (5)$$

$$\Sigma_k \leftarrow (1-\beta)\Sigma_k+\beta(c-\mu_k)(c-\mu_k)^T \quad (6)$$

The parameter α in (4) is the learning rate for the mixture components' probability. [3] uses $\beta = \alpha P(c|M_k)$, but the authors suggest that a constant may be chosen instead for performance reasons. We decided to use a constant value for β not only because of performance concerns, but since in practice such a simplification has yielded better results. In the case of a 3-channel input sequence, values for β obtained by the former method tend to be quite small relative to α. That, coupled with the liberal 2.5 threshold for a match can cause initialization problems since a mixture component that barely matches subsequent pixel values can "capture" them for a long period of time with little or no change in the mean or variance.

Some alternative approaches to choosing β appear in [4].

IV. COVARIANCE REPRESENTATION

We have postponed discussion of how the covariance matrix Σ_i is represented. Stauffer et al. used the simplifying assumption that different color channels are independent and have the same variance, thus $\Sigma_i = \sigma_i^2 I_{p\times p}$. It is possible to drop the assumption, as in [4], of uniform variance across channels and represent the covariance matrix as $\Sigma_i = \operatorname{diag}(\sigma_{r,i}^2, \sigma_{g,i}^2, \sigma_{b,i}^2)$. Lastly, it is possible to assume nothing about the covariance matrix at all.

A segmentation obtained using different covariance representations is shown in Fig. 1. The original image a) shows two cars, the topmost of which has similar colors to the road, and we can expect some holes to appear in its segmentation owing to that fact. Since b) essentially represents a match against a sphere in RGB space, it should come at no surprise that both cars' segmented images contain holes — notably the side door of the top car and the rooftop of the bottom one. Most of the holes are filled in when each channel's variation is calculated in c) where matches are performed against an axis aligned ellipsoid in RGB space. Note, however, that some pixels have been overzealously assigned to the foreground – especially those of vehicle/road boundaries. The result of using the full

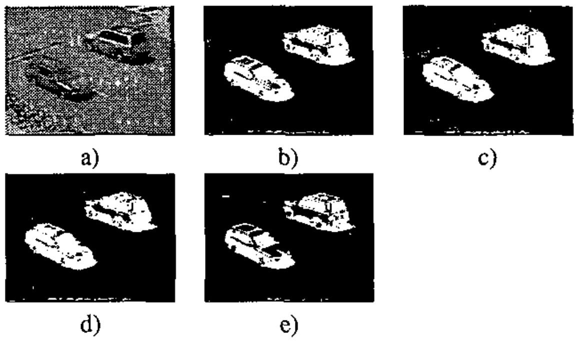

Fig. 1. The effect of using different covariance representations. a) original frame; b) independent channels, same variance; c) independent channels, each with its own variance; d) full form covariance; e) segmentation performed on the grayscale version of the image.

covariance matrix in d) combines the overall good shape recovered in b) and results in better filled regions as in c). For comparison, e) shows the segmentation obtained when using the method on a single grayscale channel — large portions of the moving objects are assigned to the background since color information is discarded.

The relative merits of using a more complete model of the covariance matrix are not very surprising. What we think is important to note is that the time to compute the segmentation in d) differs from the time to compute the segmentation in b) by a factor smaller than 2. Section 5 will detail the performance of all the methods displayed in Fig. 1.

V. PARAMETER SELECTION

A number of issues arise when appropriate parameters for the segmentation need to be selected. The desired outcome of the parameter selection is that the background properly adapts to lighting changes, objects that stay stationary for a short time do not fold into the background and that multimodal backgrounds (e.g. waving trees) are properly represented. Unfortunately a parameter selection that accounts for one desired goal is often at odds with the selection appropriate for another.

A. *Choice of number of mixture components*

The choice of number of mixture components has a direct impact on performance, even though the decrease in performance is not directly proportional to the increase in the number of components.

At the very least, 2 mixture components are necessary — one for the background and one for the foreground. Values ranging from 3 to 5 are typical, with 3 being the choice among many authors for performance reasons. We settle for 4 mixtures since there is a slight increase in quality and the performance of the system stays over the the desired frame rate.

B. Choice of color space

The mixtures of Gaussians approach has been successfully used with many different color spaces. A color space such as the HLS (Hue, Lightness, Saturation) double-cone is appropriate for the method since all dimensions are normalized. However, when using MPEG-4 compressed video results were particularly poor using HLS since the saturation component was heavily quantized in the decompressed video stream. Blocking artifacts were also extremely noticeable when using HLS.

Color spaces other than RGB may better reflect the assumption of independent channels (when modeling the covariance matrix as diagonal), but when using the full covariance matrix the correlation between the red, green and blue channels is not lost and there is no need for decoupling them by using a different color space.

C. Choice of α and β

Choosing a value close to 1 for α degenerates the method into a frame differencing approach. $1/\alpha$ controls the length of the window of past evidence considered by the algorithm, so ideally we would like the window to be long enough to prevent the inclusion of objects that stay stationary for short durations; Considering our target application of traffic intersection monitoring (where object can frequently remain stationary for up to a minute) and the desired frame rate of 30fps, α should be very small.

To guide our choice of α, we use the mixing coefficient "half-life" as a measure of how fast the background model is affected by α. The half-life is simply the time that it takes for the mixing coefficient of an unmatched mixture component to halve:

$$\mathrm{h}(\alpha) = \frac{\log_{1-\alpha} 1/2}{r} \tag{7}$$

where $\mathrm{h}(\alpha)$ is the half-life and r is the frame rate (in frames per second).

Choosing $\alpha = 0.001$ corresponds to a half-life of about 23 seconds given our desired frame rate of 30 frames per second. This choice does not impede the ability of the algorithm to cope with gradual lighting changes due to the Sun's daily motion. However, two problems arise: radical changes in the background (due to a camera shake, for example) will cause a misclassification of pixels for a large number of frames following the event; transient lighting changes (due to a passing cloud or camera gain control circuitry) will cause misclassification of pixels for the duration of the event. An easy way to combat these problems is to decrease $\mathrm{h}(\alpha)$, but that is at odds with our desire that objects do not enter the background quickly. Further, temporary lighting changes may be learned, which means that the model will misclassify pixels both at the onset and end of the lighting fluctuation. We deal with the problem of sudden changes in global lighting using a separate brightness monitor, described in sec. VI.

When using a constant value for β, it is possible to set $\beta = \alpha$, thus updating the mixing coefficients and the mixture components at the same rate. Our approach differs slightly, since we set $\beta = C\alpha$, where C is a constant. In the particular examples used throughout, we use $\beta = 1/4\alpha$, which was chosen empirically.

D. Choice of B

The choice of the background threshold parameter B is somewhat arbitrary. Selecting a value too close to 0 means that only a single mixture component will be considered a part of the background process, counter to the desire to allow multimodal background processes. Selecting a value close to 1 results in objects becoming part of the background too quickly and may in fact prevent even a single mixture component from representing the foreground process. Between these undesirable extremes, the choice depends to a large extent on the nature of the scene. We favor a threshold of 0.4–0.5 rather than larger values suggested by other authors since in a traffic intersection scene there are regions of high activity where the background is not visible too often due to the regular traffic cycle. Note that while the same effects can be achieved be either modifying α or B, α is the parameter that has a meaningful interpretation in terms of a time quantum. That is to say that α should be chosen to reflect some time-related characteristics of the scene (average wait at a traffic light), and B should be adjusted afterwards in accordance with our expectation of how "busy" regions in the scene can get.

E. Initializing mixture components

The last set of parameters that must be chosen are the initial mixing coefficient of mixture components added to the model and the initial variance of said mixtures. Our particular implementation also allows for a low threshold on the covariance to be set. All initialization and threshold values related to the covariance matrix are per-channel.

The initial weight given to a new mixture component controls how fast it will be allowed to enter the background after being observed for the first time. The intuitive meaning of the initial weight is that if set higher than α, mixture components can enter the background faster than they can leave it due to lack of evidence. If set lower than α, the initial weight prevents the newly initialized mixture to enter the background as quickly as it is allowed to leave it due to lack of supporting evidence. We simply choose to use α as the initial value for the weight since that is the actual mixing weight that a mixture component would have obtained if a weight of zero was updated according to (4).

The choice of the initial variance for a mixture component is important given the very slow rate at which the variance is allowed to change in the long run. Since the assignment of observed pixel values to mixture components is discrete, and the number of mixtures is fairly limited, it is inappropriate to initialize the variance too high and depend that it will eventually reach its "true" value. In

Parameter	Symbol	Value	Mutable
Number of Gaussians	N	4	no
Learning rate	α	10^{-3}	yes
Learning rate	β	$\alpha/4$	yes
Match threshold	δ	2.5	yes
Background proportion	B	0.5	yes
Initial weight		α	yes
Initial variance		320.0	yes
Low variance threshold		49.0	yes

Fig. 2. Parameter values used in examples.

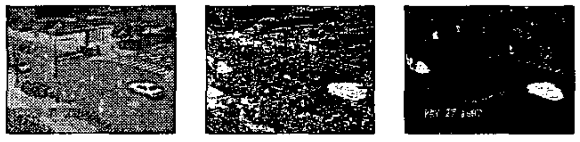

Fig. 3. Effect of sudden brightness change. a) original frame; b) The segmentation obtained without monitoring brightness and contrast; c) Segmentation obtained by rate-limiting brightness and contrast changes.

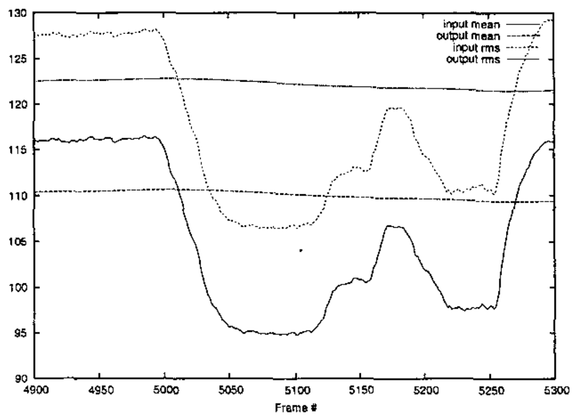

Fig. 4. Input and output mean and root-mean-squared after applying the method to the sequence from which the scene in Fig. 3 was chosen.

selecting the value, we make the observation that under normal circumstances, we would expect the range of pixel values that match a given mixture component to be less than the range one would use if a fixed background substraction was being performed. Thus, if F is an appropriate threshold for use in known background substraction or inter-frame differencing, we can use $(F/\delta)^2 I_{p\times p}$ as a reasonable initial value for Σ.

The last parameter that needs to be chosen is a low threshold below which the variance should not be allowed to fall. Such a threshold is necessary since the input color values are discrete — a variance with magnitude less than 1, for example, makes no sense if the possible deviations are integral. We can choose the low variance threshold similarly to the choice of initial variance, by selecting a threshold below which differences in RGB values are imperceptible and converting to a corresponding variance threshold.

F. Summary of parameters

The table in 2 shows the parameter values chosen for our examples and the subsequent performance analysis. Mutable parameters can change in value from frame to frame, while immutable parameters must be chosen in advance since their modification is impractical at run-time. As suggested in [2], some of the parameters may have different values at the start of processing to facilitate the proper initialization of the background.

VI. BRIGHTNESS EQUALIZATION

The need for monitoring the brightness of the incoming video in an outdoor scene is succinctly illustrated by Fig. 3. As a bright white car enters the scene (a), the camera's exposure rapidly changes and causes a decrease in brightness and contrast. The result is a misclassification of a significant number of image pixels (b). As c) of Fig. 3 shows, the segmentation of the same frame performed after brightness and contrast correction is clean and reflects the moving objects in the scene. The one disadvantage is that the date display overlaid by the camera onto the image has a constant color regardless of the scene, so the modification causes it to temporarily enter the foreground. This is not a big issue since the date rarely coincides with the region of interest in the image.

Let $\mathcal{I}(x,y,t)$ is the input frame at time t. A new image, $\mathcal{O}(x,y,t) = A\mathcal{I}(x,y,t)$ is computed by applying a linear transformation to all n pixels of $\mathcal{I}$ so that the brightness and contrast of $\mathcal{O}$ match the target brightness b_t and contrast c_t. The image $\mathcal{O}$ becomes the input to the background segmentation module, while b_{t+1} and c_{t+1} are calculated using the brightness and contrast measure of $\mathcal{I}$.

We use the mean color of $\mathcal{I}$ as a measure of its brightness and the root-mean-squared of $\mathcal{I}$'s colors as a measure of contrast (brightness affects the root-mean-squared as well). Given $\bar{b} = \frac{1}{n}\sum_{(x,y)} \mathcal{I}(x,y)$ and $\bar{c} = \sqrt{\frac{1}{n}\sum_{(x,y)} \mathcal{I}(x,y)^2}$ we must choose the transformation A so that

$$A\begin{pmatrix} \bar{b} \\ \bar{c} \end{pmatrix} = \begin{pmatrix} b_t \\ c_t \end{pmatrix} \tag{8}$$

After $\mathcal{O}$ is computed, we combine b_t and c_t with $\bar{b}$ and $\bar{c}$ to get

$$b_{t+1} = (1-\beta)b_t + \beta\bar{b} \tag{9}$$

$$c_{t+1} = (1-\beta)c_t + \beta\bar{c} \tag{10}$$

The update factor β is the same as that used for updating the mean and variance of pixels in the background model. Intuitively, the brightness and contrast are not allowed to change faster than the background model can incorporate the change in the input due to global lightning changes.

The sudden change in brightness in Fig. 3 is visible as a downward spike in both the mean and root-mean-squared of the input sequence, shown in Fig. 4. The input presented to the background segmentation module reveals only a slight

	No filter	With filter
a)	47s (63.8 fps)	49s (61.2 fps)
b)	55s (54.5 fps)	58s (51.7 fps)
c)	69s (43.5 fps)	72s (41.7 fps)

Fig. 5. Performance of method with and without brightness and contrast compensation. a) independent channels, same variance; b) independent channels, separate variances; c) full covariance matrix.

decrease in the brightness and contrast measure, which is accommodated by the background without problems as c) in Fig. 3 demonstrates.

The brightness and contrast control method described in this section performs best when the camera observing the scene is sufficiently far away that moving objects do not fill a substantial portion of the image. The appearance of a very dark or bright object can influence the chosen brightness and contrast measures even in the absence of global illumination changes. An area of future work is to incorporate the segmentation results from a previous frame in order to compute these measures only for areas of the image that are highly likely to be background. Another approach under consideration is to compute the brightness transformation for each one of a number of smaller image regions (most likely tiles) and apply a consensus transformation that is consistent with global brightness and contrast changes — brightness and contrast changes caused by moving objects will be limited to a small number of tiles and such outliers will not affect the choice of transformation parameters.

VII. PERFORMANCE

Figure 5 shows the running time and corresponding frame rate obtained using the three different covariance matrix representations, with and without the brightness and contrast filter. The input video sequence has resolution 320×240. The tests were run on a 2.66GHz Pentium IV machine on a sequence of 3000 frames. The methods were allowed to run for 1000 frames before measurements were taken since under our implementation the methods run slightly faster when only a small number of mixture components are initialized. The overhead of decompressing the video sequence (from MPEG-4) is included in the reported times. The variation in the running times across different runs was about 2s, mostly due to to the low resolution timer used to measure the times.

The performance results show that the mixture of Gaussians method is perfectly capable of running in real-time, even if no simplifying assumptions about the covariance matrix is made. The brightness and contrast filter imposes a small performance penalty on the background segmentation process, but simplifies parameter tuning (by eliminating one factor that influences parameter choice).

A. Implementation Notes

The running time of our traffic monitoring application is dominated by the time required to perform the background segmentation. Some freely available code for implementing a mixture of Gaussians method are available, but their performance was unsatisfactory for images of quarter-VGA resolution and full 30fps frame rates. We believe that such performance problems are not necessarily due to fundamental problems with the method, but because the implementations did not focus on computational performance. Several implementation techniques helped improve the performance of the algorithm as presented here.

The first improvement in our implementation is based on the realization that the mixing coefficients do not need to be renormalized after update at all, since knowing their sum is sufficient to allow proper classification into background and foreground relative to the threshold B. Individual mixing coefficients are always in the range $[0..1]$ so no special overflow checks are necessary.

A further optimization is an early-exit strategy: as soon as a match against a mixture component is determined, there is no need to calculate the distance between the pixel value and less likely mixture components. Only the mixing coefficient of the remaining mixture components needs to be updated. Also, regardless of the covariance matrix representation used, it is possible to use some of the temporary results calculated during the check for a match against a mixture component in the update equations. Since the check and parameter update portions of the code are the hot-spot, it pays to rewrite all equations so as to minimize the number of arithmetic operations used (even fast ones like addition and multiplication).

Since non-matched mixture components do not change their relative order, at any step only the matched (or newly initialized) mixture component could potentially move in the likelihood ordering of all components. The rarity of such reorderings was used to guide the structure of the code that maintains mixture components in sorted order.

Regardless of the covariance representation used, the operation of the algorithm requires only addition, substraction and multiplication operations (with suitable rewriting of some tests involving ratios); this provides significant gains on modern Intel architectures since these floating point operations are fully pipelined.

Single-precision floating point values are sufficient to represent all quantities involved in the algorithm due to the discrete nature of the input data — in fact the classification computed using single and double precision numbers was found to be bit-identical for a large input sequence.

Lastly, all data pertaining to individual mixtures is kept together in memory to improve the locality of memory references and wherever possible conditional statements were replaced with equivalent bitwise arithmetic expressions to avoid pipeline stalls in the CPU.

VIII. CONCLUSIONS

We described a practical implementation of the mixture of Gaussians method for background segmentation. We achieved full frame rate on 320×240 pixel input video sequence without sacrificing accuracy. We further demonstrate that the performance differential between the use of a diagonal and a full covariance matrix in the individual mixture components' parameters is small. The latter is significant since it allows the use of plain RGB input sequences without sacrificing quality due to channel correlation and makes better use of the available color information. The selection of the method's parameters is examined and specific sample values are provided for all of them.

A practical online method for coping with sudden brightness changes was described. The method pre-filters the input and thus does not necessitate any changes to background substraction code.

IX. ACKNOWLEDGMENTS

This work has been supported in part by the Minnesota Department of Transportation, the ITS Institute at the University of Minnesota, and the National Science Foundation through grants #CMS-0127893 and #IIS-0219863.

REFERENCES

[1] Veeraraghavan H., Masoud O., and Papanikolopoulos N.P. "Computer Vision Algorithms for Intersection Monitoring," *IEEE Trans. on Intelligent Transportation Systems* vol. 4, no. 2, pp. 78–89, Jun. 2003.

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

[3] Stauffer, C. and Grimson, W. "Adaptive Background Mixture Models for Real Time Tracking." *Computer Vision and Pattern Recognition.* 1999.

[4] Power, W. and Schoones, J. "Understanding Background Mixture Models for Foreground Segmentation" *Proc. Image and Vision Computing.* 2002.

[5] Gutchess, D., Trajkovic, M., Cohen-Solal, E., Lyons, D., and Jain, A. K. "A Background Model Initialization Algorithm for Video Surveillance." *In Eighth International Conference on Computer Vision.* vol. 1, pp. 733–740, 2001.

[6] Dar-Shayng Lee, Hull, J., Erol, B. "A Bayesian Framework for Gaussian Mixture Background Modelling." *Proceedings of ICIP 2003.* vol. 3, pp. 973–976, 2003.

[7] Bevilacqua, A. "Effective Object Segmentation in a Traffic Monitoring Application." *3rd IAPR Indian Conference on Computer Vision, Graphics and Image Processing.* 2002.