LOW-COMPLEXITY AND HIGH-EFFICIENCY BACKGROUND MODELING FOR SURVEILLANCE VIDEO CODING

Xianguo Zhang, Yonghong Tian, Tiejun Huang, Wen Gao
National Engineering Laboratory for Video Technology, Peking University, Beijing 100871, China
{xgzhang, yhtian, tjhuang, wgao}@pku.edu.cn

ABSTRACT

[1] Recently, background modeling (shortly BgModeling) plays a more and more important role in high-efficiency surveillance video coding. Meanwhile, many practical video coding applications also present some specific requirements for BgModeling, such as the low memory cost and low computational complexity. However, existing BgModeling methods are mostly designed for video content analysis such as object detection. Thus they may be not directly applicable for video coding. In this paper, we firstly present an analysis for the features of BgModeling in surveillance video coding and make a comparison of the performances of existing BgModeling methods. Then we propose a segment-and-weight based running average (SWRA) method for surveillance video coding. SWRA firstly divides pixels at each position in the training frames into several temporal segments, and then calculate their corresponding mean values and weights. After that, a running and weighted average procedure is used to reduce the influence of foreground pixels and finally obtain the modeling results. Experimental results show that, the SWRA-based encoder achieves the best performance over several state-of-the-art methods, with much less cost of memory and modeling time.

Index Terms**—**background modeling, surveillance video coding, Gaussian mixture model, complexity, memory

1. INTRODUCTION

As surveillance cameras have been widely deployed in many surveillance systems, it is highly desirable for developing low-complexity and high-efficiency surveillance video coding methods. In general, the recent video coding standards such as MPEG-4, H.264/AVC and HEVC can provide relatively high coding efficiency. However, these standards the optimal choices for surveillance video, since they are basically designed for general video applications such as video broadcasting. Therefore, it is attractive to take the special characteristics into account to develop novel coding schemes for surveillance video.

Typically, most of surveillance cameras are usually deployed on a fixed position and often used to capture the scene at a certain direction for a long time. In this case, a background frame can be generated and updated periodically. Using the background as the prediction reference, background modeling (BgModeling) is highly beneficial for a high-efficiency surveillance video coding.

In general, existing BgModeling methods can be roughly classified into two categories: parametric methods such as Gaussian mixture model (GMM) [1-3], and nonparametric methods including Bayesian modeling, kernel density estimation, temporal median filter, mean-shift [4] and etc. Using these parametric and non-parametric methods, some recent studies in [5-7] have introduced several very efficient and practical methods for efficient surveillance video coding and transcoding. Typically, these works follow the BgModeling based coding framework shown in Fig. 1.

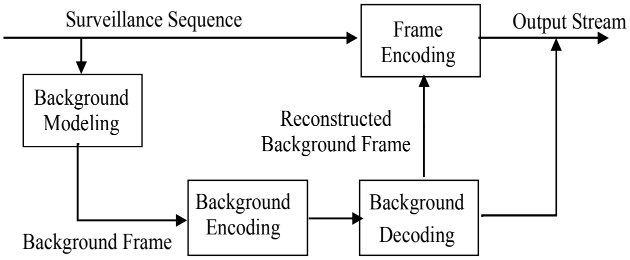


Fig.1 Framework of BgModeling based encoders

Actually, existing BgModeling methods are mostly designed for video content analysis such as object detection [4], where the main objective of BgModeling is to capture the static properties of a scene such that the foreground objects can be easily detected. However, the situation is different in the surveillance video coding framework, because the background model for video coding should be high-quality encoded with a mass of bits and only updated and encoded once in a long period. As analyzed in section 2, BgModeling in surveillance video coding has many remarkable features and requirements, including periodically updating, small prediction residual error variance, low computational complexity and low memory cost, and no-delay modeling.

In [8], Piccardi presents a conclusion that the running Gaussian average and the median filter methods have a low computing complexity and limited memory requirements, but offer the comparable accuracy while achieving a high frame rate. This suggests a low-complexity but high-efficient BgModeling algorithm is feasible in surveillance video coding. Therefore in this paper, we firstly present an

[1] Prof. Tiejun Huang is the corresponding author.

analysis of the features that BgModeling should meet in surveillance video coding. Meantime, using different BgModeling methods, a detailed performance comparison and analysis are given. Motivated by these analysis results, we propose a segment-and-weight based running average (SWRA) method. SWRA firstly divides pixels at each position in the training frames into several temporal segments, and then calculates their corresponding mean values and weights. Afterwards, a running and weighted average procedure is used to reduce the influence of foreground pixels and obtain the modeling results.

By using temporally segmenting and weighted averaging, our method can reduce the influence of the training frames' foreground pixels during the BgModeling process. As a result, it can effectively meet the requirements of lower complexity and less memory cost. Experiments show that the SWRA-based encoder achieves the best performance over several state-of-the-art methods, with much less cost of memory and modeling time.

The rest of this paper is organized as follows. Sec. 2 discusses features for BgModeling in surveillance video coding and presents a performance and complexity analysis for common methods. Sec. 3 introduces the algorithm. Experiments and conclusion are introduced in Sec. 4 and 5.

2. ANALYSIS

2.1. Features of BgModeling in surveillance video coding

In BgModeling for object detection, as referred in [5], supposing there is an ideal background C', system noise N_{sys}, moving object M_{obj} and a moving background M_{bgd} in a scene, we formulate the observed values of scene V_{obsv} by:

$$V_{obsv} = C' + N_{sys} + M_{obj} + M_{bgd}\,, \qquad (1)$$

where the symbol '+' denotes the cumulative effect. Then the ideal background C_i' for the i-th frame is derived by

$$C'(i) = V_{obsv}(i) - N_{sys}(i) - M_{obj}(i) - M_{bgd}(i). \qquad (2)$$

We can see from (2) that, the background frame suitable for object detection should be constantly updated utilizing the original input frame. However, that is not feasible in surveillance video coding system, because each updated background frame should be encoded into stream again to guarantee the decoding match. To avoid the burst bit-rate increase caused by updating the background frame, surveillance video coding framework in Fig. 1 updates the background frame once in a long period (namely *LGOP*), and frames in the next *LGOP* utilizes the last reconstructed background frame as prediction reference. Consequently, the best background frame *Bg* in surveillance video coding for the *n* frames in the next *LGOP* satisfies:

$$Bg \;=\; \arg\min_{b} \left\{ \sum_{i=0}^{n} J(b,\; V_i^{obsv}) \;\middle|\; b \in B \right\}, \qquad (3)$$

where b is arbitrary modeling background frame in the available set of background frames B, and $J(b,\ V_i^{obsv})$ is a function for calculating the rate-distortion cost of encoding the i-th observed scene V_i^{obsv} with b as reference. Because *Bg* is only updated after an *LGOP*, the first feature is concluded as *periodically background updating*.

A classical video coding theory for motion compensated prediction in [9] states that, the prediction error variance reflecting the power spectral density determines the coding efficiency of a reference frame at the same MSE. In surveillance video coding, as referred in [5-7], the background frame is mainly used as a long-term reference for the following input frames. Thus the second feature is *small prediction residual error variance*. The 3rd feature is *low computational complexity and low memory cost*: Buffered memory used in BgModeling procedure should not be very large in video coding, especially in hardware and parallel systems. Moreover, complex operations (e.g. multiplication and division) and high-precision data structure (e.g. float and double) should be avoided as much as possible. Besides, video surveillance is always a no-delay system, and this requires the modeling algorithm must be *no-delay modeling*. That is, read each input frame only once.

2.2. Efficiency and complexity of BgModeling for coding

To evaluate the efficiency of different common BgModeling methods for surveillance video coding, four typical methods are implemented and embedded into H.264/AVC baseline profile encoder (BP). The four methods are the Mean-Shift (namely MS) proposed in [4], the popularly used Gaussian running average (RA), and the Gaussian Mixed Models [3] using 1 or 5 models for each pixel (GMM-1 or GMM-5). Corresponding Encoders are namely BP-MS, BP-RA, BP-GMM-1 and BP-GMM-5. The encoding results on different CIF&SD surveillance sequences can be seen from Table 1. As is shown, comparing with the BP encoder without BgModeling, BP-GMM-5 obtains the highest performance.

Table 1. BgModeling based BP vs. BP on PSNR gain (dB)

BP-x on SD	*Crossroad*	*Overbridge*	Bank	*Office*	**average**
GMM-1	0.92	1.37	1.13	0.40	0.96
RA	1.22	1.73	1.71	0.58	1.31
MS	1.26	1.81	1.68	0.74	1.37
GMM-5	1.34	1.94	1.79	0.88	1.49
BP-x on CIF	*Crossroad*	*Overbridge*	*snowgate*	*snowroad*	**average**
GMM-1	0.79	0.50	1.13	0.91	0.84
RA	0.93	0.80	1.75	1.34	1.21
MS	1.01	0.89	1.62	1.23	1.19
GMM-5	1.22	0.95	1.76	1.51	1.36

Table 2. Memory cost for each pixel (byte)

ITEM	**RA**	**GMM-1**	**GMM-5**	**MS**
Buffered frames	1×S(char)	1×S(char)	1×S(char)	M×S(char)
Mean values	1×S(float)	2×S(double)	2×S(double)	1
Weight	0	1×S(double)	1×S(double)	0
Threshold/variance	0	1×S(double)	1×S(double)	0
Match points	0	1×S(char)	1×S(char)	0
SUM of	**5**	**34**	**34×5 = 170**	**M=120**

Table 3. Modeling time on different sequences (second)

CIF	*Crossroad*	*Overbridge*	*snowgate*	*snowroad*	**average**
RA	1.6	1.7	1.7	1.7	1.7
GMM-1	5.9	5.9	5.8	5.8	5.9
GMM-5	11.5	11.3	10.4	10.3	10.8
MS	61.8	61.2	57.0	56.4	59.1
SD	*Crossroad*	*Overbridge*	*Office*	*Bank*	**average**
RA	6.6	6.6	6.7	6.6	6.6
GMM-1	24.4	24.5	24.5	24.4	24.5
GMM-5	43.3	41.8	46.4	41.0	43.1
MS	242.8	236.2	252.8	235.9	241.9

For BgModeling in surveillance video coding, as is referred, performance, memory cost and running time are the same important factors. The calculation for their memory cost in each pixel position is listed as follows.

(1)RA: one current pixel with type of char and one float-precision mean value for each pixel should be buffered.

(2)GMM-X: besides the buffered input pixel, a GMM model is required to be buffered. The model is composed of double-precision mean value, variance and weight. Moreover, an 8-bit value should be stored to count the number of matched points for each GMM model.

(3)MS: Mean-shift based algorithms usually buffer all the training frames and very few additional temporal variables are used for the clustering and sorting operations.

Supposing the number of training frames is denoted by *M=120*, from the above analysis of RA, GMM-X and MS, the memory cost for each algorithm is listed in Table 2.

As for the modeling time, Table 3 gives the detailed information for each algorithm. This result indicates MS owns the largest modeling time and GMM-5 spares much more time than RA. Moreover, it can be concluded that RA works without dependency on scene texture, and other methods are sensitive to sequence content.

In a brief summary, GMM-5 contributes largest to video coding performance gain but spares a relative large memory and time cost. In practical system, especially in parallelism or hardware environment, such GMM-5 cannot meet the requirement for fast modeling and low memory cost. This inspires us to propose a method which can achieve higher performance with less memory and time cost.

3. PROPOSED BGMODELING METHOD

As analyzed in Sec. 2, BP-GMM-5 obtains the best performance, but consumes largest memory and spares much background modeling time. Therefore, we engage to accomplish a new background model which can save the cost and maintain the background quality.

To maintain or improve background quality, an ideal solution is to calculate the mean value of all the background pixels in the training frames. However, it is very difficult in recent years to exactly evaluate which pixels belong to the background. Physically, "background" equals to the most frequently appearing content. This inspires us to propose a segment-and-weight based running average (SWRA) to approximately calculate the mean value of background pixels. As SWRA is based on a running average procedure, there will not be large memory cost and computational complexity. Generally, SWRA divides the pixels at a position in the training frames into temporal segments with their own mean values and weights, and then calculates the running and weighted average result on the mean values of the segments. In this procedure, pixels in the same segment have the same background/foreground property, and we just set the long segments with much larger weight. This mechanism ignores the foreground/background property of each segment, so *low memory cost* and *no-delay modeling* are guaranteed. Besides, the running average using weights satisfy the *low computational complexity* requirement. Experimental results show it maintains performance and generates *small prediction residual error variance.*

In detail, as the five steps of the algorithm shows in Fig. 2, SWRA initialize the parameters, calculates the threshold, divide segments, calculates the mean value and models a background value of pixels at arbitrary position (x, y).

(1) *Initialize*: Initialize background model value *AVG* and its weight *W* for the following weighted average procedure to 0, and then create first segment. Length of the first segment *L* equals to 0 and its mean value *avg*=0. The model value before the current segment *avg*' is also set 0.

```
Input:
  T={$I_i$=$f_i(x,y)$| i=1~N },where $f_i(x,y)$ is a frame in the N training frames.
Modeling:
  For each position (x,y), Begin
      {L, avg, avg', AVG, W}=0, $Th_0$=14··················  1)
      While i =1~ N, Begin
        if (x, y)=(0, 0) , Begin              ··················  2)
            $Th_i = 2\sqrt{Round\left(\left(\sum_{x,y}\left(Cmp(x,y)\times Diff(m,n)\right)\right)/Sum\right)}.$
            $where\ Sum=\sum_{m,n}\left(Cmp(m,n)\right),$
                $Cmp(m,n)=\begin{cases}1 & if\ Diff(m,n)\le Th_{i-1}\\0 & if\ Diff(m,n)>Th_{i-1}\end{cases}$
                $and\ Diff(m,n)=\left|I_i(m,n)-I_{i+1}(m,n)\right|$
        End
            $L=\begin{cases}L+1, & if\ |(I_i(x,y)-I_{i+1}(x,y))|<Th_i\\L, & if\ |(I_i(x,y)-I_{i+1}(x,y))|\ge Th_i\ and\ L>N/20\\0, & L\le N/20\end{cases}$  3)
        if $|(I_i - I_{i+1})|< Th_i$     $avg=\left(Round\left(avg\times L+I_{i+1}\right)\right)/L$  4)
        else if $L>N/20$     $avg=0,\ W=0$
        else
            $W=W+L^2,\ avg'=Round\left(\left(avg'\times W+L^2\times avg\right)/\left(W+L^2\right)\right)$
      End
      $Bg(x,y)=AVG=Round\left(\left(avg'\times W+L^2\times avg\right)/\left(W+L^2\right)\right)$  5)
  End
Output: Background frame Bg
```

Fig.2 Algorithm of the proposed SWRA

(2) *Calculate the threshold for segmenting*. The normal distribution theory states as follows

$$f(x)=\frac{1}{\sqrt{2\pi}\,\sigma}e^{\frac{(x-\mu)^2}{2\sigma^2}}, \qquad (4)$$

where μ is the mean value, σ^2 is the mean square error. Because the probability of $|f(x)-\mu|>2\sigma$ is less than 4%, so we use 2σ as the threshold for *th* temporally segmenting a pixel in training frames. To reduce the memory cost, the threshold *th* is initialized as 14, and updated in this step as follows:

```
For each (i, j) in the $k^{th}$ frame $I_k$, Begin
        $sum = 0$, $num = 0$
        $d = | I_k(i,j) - mean|$.
        if  $d < th$,
            $sum = sum+d$, $num = num+1$.
 End
$Th_k = 2\times ( Round(sum/num) )^{1/2}$
```

Fig. 3 Threshold updating

(3) *Create a new segment or widen the current segment.* At arbitrary position, a new temporal segment will be created if the *d* in Fig. 3 is larger than *th*. Otherwise, length of the current segment is widened. Through this procedure, temporally successive pixels at arbitrary position can be divided into segments, as shown in Fig. 4. Borders between segments stand for a texture switch which occurs on adjacent frames. Note that, if the length of the current segment is too short, the weight of much shorter segment is set 0. Practically, 1/20 of the training length is used to judge texture switch.

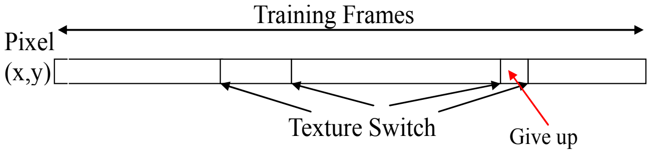


Fig.4 Diving training frames into segments

(4) *Calculate mean value and weight for each segment*. The weight of each segment is set square of its length. Besides, a running average procedure will be employed for *"low computational complexity"*. Supposing length and mean value of segment *k* are *len*$_k$ and *avg*$_k$, Fig.4 can be rewritten by:

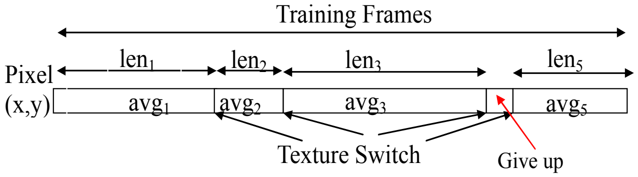


Fig.5 calculates mean value and weight for each segment

(5) *Generate and output the background value*. In a practical system, to satisfy *"low memory cost,"* we do not buffer the length and mean values of each segment. Instead, we just interactively buffer and calculate the total mean value *AVG* and its weight *W* from 1st to *k*th segment by

$$\text{AVG}= (\text{AVG}\times \text{Weight}+\text{len}_k^2\times \text{avg}_k) / (\text{Weight}+\text{len}_k^2) \qquad (5)$$

$$\text{Weight} = \text{Weight}+ \text{len}_k^2 \qquad (6)$$

Such calculation procedure is shown in Fig. 6. It indicates we only need to buffer and derive the *AVG* and *W* of the first *k* segments from the first *k-1* segments. Following this, when the current segment reaches the end of training frames, we will calcuate the final *AVG* and *W*. At last, we will obtain the required background frame by jointing the *AVG* of each pixel together.

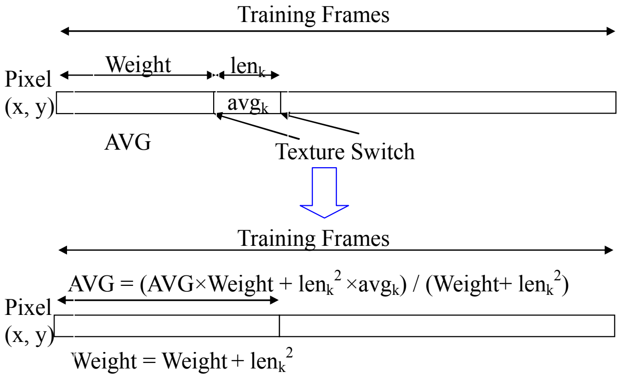


Fig.6 The calculation of buffered AVG and W

From the above statement, the proposed SWRA works based on weights and running average. Thus it will successfully satisfie the features of *no-delay modeling*, *low complexity and memory cost*. Besides, SWRA updates the model value of each pixel at the end of a segment, and background frame is updated at the end of training frames. As a result, SWRA satisfies the demand of *periodically background updating*. In Sec.4, we will practically evaluate the efficiency, memory and time cost of SWRA.

4. EXPERIMENTS

4.1 Experimental Setup

Traditionally, PSNR and bit-rate are used as metrics for video coding efficiency. To evaluate the efficiency of the surveillance video coding using SWRA, H.264/AVC High profile encoder (HP) and four other HPs are employed as anchors for comparison. The anchors include the Gaussian mixed model using 1 and 5 models for each pixel (GMM-1 and GMM-5), the mean-shift(MS) and the Gaussian running average (RA). Encoders using the methods are respectively namely HP-GMM-1, HP-GMM-5, HP-MS and HP-RA.

For an undisputed comparison, all the five anchors and SWRA are implemented on JM17.2 (believable H.264/AVC reference software). JM is configured by test conditions in the officially accepted proposal [11] as shown in Table 4. Moreover, background frames in encoders for the same sequence are updated and encoded simultaneously.

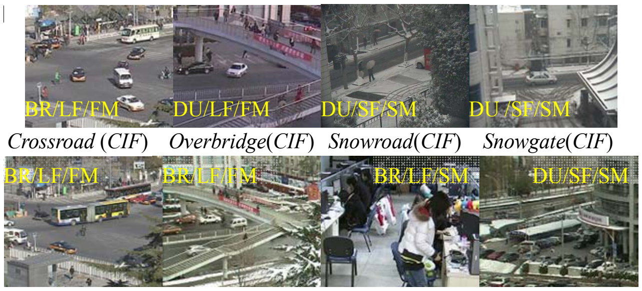
Table 4. Configuration for JM17.2

Item	Descr.	Item	Descr.
Ref. Num.	5	QP of I frame	22,27,32,37
IntraPeriod	0	Entropy Coding	CABAC
Profile/Level	High	8x8Transform	Enable
Long-term	Enable	Adaptive Rounding	Enable
RDO Quant.	Used	SAD Method	hadamard
Motion Esti.	UMHexgon	B frames Number	2
1/4-pel ME	Enable	Search Range	32
Modes	ALL	RDO	Used
QP of P frame	23,28,33,38	QP of B frame	24,29,34,39

4.2 Test Sequences

The high-quality and large-bits encoded background frame will add large bits into stream, but offers a very effective reference frame for follow-up frames. With such background as reference (QP for encoding background frame is set 0), there will be more bits to be saved. Consequently, a believable experiment to evaluate BgModeling algorithms requires long sequence for testing. Because of this, eight sequences with more than 1000 frames from AVS workgroup [10], as shown in Fig. 7 and Table 5, are encoded by HP-MS, HP-RA, HP-GMM-1 and HP-GMM-5. Resolution of these eight sequences is from CIF to SD, and the scenes include sunny and dusky (BR/DU), large and small foreground (LF/SF), fast and slow motion (FM/SM).

As is seen, Crossroad(SD), Overbridge(SD), Office(SD) and Crossroad(CIF) are more bright than others, Crossroad(SD), Overbridge(SD), Office(SD) and Crossroad(CIF) and Overbridge(CIF) owns large size of motions and large proportion of foreground pixels.



Crossroad (*CIF*) *Overbridge*(*CIF*) *Snowroad*(*CIF*) *Snowgate*(*CIF*)

Crossroad (*SD*) *Overbridge* (*SD*) *Office* (*SD*) *Bank* (*SD*)

Fig.7 Test sequences and their content

Table 5. Description of test sequences

Sequence	Resolution	Brightness	Motion	Foreground Portion
Bank	720×576	dusk	small	Low Percent
Crossroad	720×576	**bright**	**large**	**High Percent**
Overbridge	720×576	**bright**	**large**	**High Percent**
Office	720×576	**bright**	**large**	**High Percent**
snowroad	352×288	dusk	small	Low Percent
Crossroad	352×288	**bright**	**large**	**High Percent**
Overbridge	352×288	dusk	**large**	**High Percent**
snowgate	352×288	dusk	small	Low Percent

4.3. Results

In this part, encoding performance of HP-RA, HP-MS, HP-GMM-1, HP-GMM-5 and the proposed HP-SWRA are firstly compared in 4.2.1. Afterwards, a detailed comparison is presented in 4.2.2 and 4.2.3 about the memory cost and modeling time among RA, MS, GMM-1 GMM-5 and proposed SWRA. Results show that SWRA can help to achieve the best performance over other BgModeling algorithm in average. Moreover, compared to GMM-5, it only consumes 10% of the memory cost and spares 25% of GMM-5's modeling time.

4.3.1 Encoding Performance

Compared with the basic HP encoder, video coding performance gain of HP-RA, HP-MS, HP-GMM1, HP-GMM5 and HP-SWRA can be seen from Table 6. It indicates that HP-GMM-X encoders seriously rely on the number of models utilized for each pixel. HP-GMM-1 achieves a much worse performance than other BgModeling algorithms, and HP-GMM-5 achieves better performance than HP-RA, HP-MS and HP-GMM1. In average, HP-SWRA achieves the best performance at 1.197/1.23 dB gain over HP on CIF/SD sequences. SWRA is slightly better than HP-GMM-5, which achieves 1.197/1.22 dB gain. Besides, HP-MS is proved more efficient than HP-RA in LF sequences, but less efficient in SF ones. Rate-Distortion-curves with snowgate(CIF) as example is shown in Fig. 8.

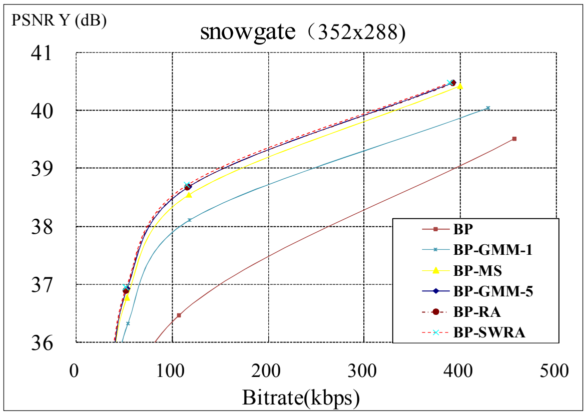


Fig.8 Rate-Distrotion curves of different methods for snowgate

Table 6. BgModeling based HP vs. HP on PSNR gain (dB)

HP-x on SD	*Crossroad*	*Overbridge*	*Bank*	*Office*	**average**
GMM-1	0.65	1.37	0.67	0.08	0.694
RA	0.93	1.73	1.24	0.30	1.052
MS	0.96	1.81	1.22	0.41	1.097
GMM-5	1.02	1.94	1.32	0.51	1.197
SWRA	1.07	1.93	1.33	0.46	1.199
HP-x on CIF	*Crossroad*	*Overbridge*	*snowgate*	*snowroad*	**average**
GMM-1	0.55	0.26	1.26	0.81	0.72
RA	0.72	0.56	1.89	1.28	1.11
MS	0.78	0.61	1.77	1.17	1.08
GMM-5	0.93	0.65	1.89	1.41	1.22
SWRA	0.90	0.68	1.95	1.38	1.23

Table 7. Memory cost (byte) for each pixel in BgModeling

ITEM	RA	GMM-1	GMM-5	MS	SWRA
Buffered pixel	1×S(char)	1×S(char)	1×S(char)	M×S(char)	1×S(char)
Mean values	1×S(float)	2×S(double)	2×S(double)	1	2×S(float)
Weight	0	1×S(double)	1×S(double)	0	2× S(char)
Threshold	0	1×S(double)	1×S(double)	0	1×S(float)
Match points	0	1×S(char)	1×S(char)	0	0
Total	**5**	**34**	**34×5 = 170**	**M=120**	**14**

4.3.2 Memory Cost

Memory cost calculation for RA, MS, GMM-1 and GMM-5 has been referred in Sec.2. In further for SWRA, the required memorized data for each pixel include: one current pixel with type of char and one float-precision mean value; two float-precision mean values *avg'*/*avg* and their corresponding char-type weights. In summary, the total memory cost is 14 bytes for each pixel. The memory cost comparison is summarized in Table 7, where *M* is the number of training frames. It shows that, RA only needs 5 bytes for each pixel, which is the least. Proposed SWRA spares much less than GMM-X and MS, and GMM-5 requires 170 bytes memory for each pixel. Note that, *S(double)* means the number of bytes in type *double* .

4.3.3 Background modeling time

The modeling time comparison results for different sequences are shown in Table 8. The RA spares the least modeling time and MS spares the largest. Moreover, it shows that SWRA spare much less time than MS, GMM-1 and GMM-5 on all the sequences, only about 25% of the computing time used by GMM-5. Moreover, SWRA is not very sensitive to sequence content, which is quite different from GMM-X and MS.

In a brief summary, the proposed SWRA can facilitate a high-performance surveillance video encoder with less memory cost and modeling time.

Table 8. Modeling Time Comparison (second)

CIF	***Crossroad***	***Overbridge***	***snowgate***	***snowroad***	**average**
RA	1.6	1.7	1.7	1.7	1.7
GMM-1	5.9	5.9	5.8	5.8	5.9
GMM-5	11.5	11.3	10.4	10.3	10.8
MS	61.8	61.2	57.0	56.4	59.1
SWRA	2.2	2.3	2.3	2.3	2.3
SD	***Crossroad***	***Overbridge***	***Office***	***Bank***	**average**
RA	6.6	6.6	6.7	6.6	6.6
GMM-1	24.4	24.5	24.5	24.4	24.5
GMM-5	43.3	41.8	46.4	41.0	43.1
MS	242.8	236.2	252.8	235.9	241.9
SWRA	10.7	10.4	10.5	10.3	10.5

5. CONCLUSION

In this paper, we analyze the main features of background modeling algorithms used in surveillance video coding and make a comparison among the common methods. Following the comparison result, we proposed a higher-performance, low-memory-cost and less-modeling-time SWRA method. The basics of SWRA can be summarized by: (1)divide the values of a pixel in training frames into segments, (2) calculate weight and mean value for each segment, and (3)a running average procedure is employed on these mean values using the weights. Experiments for comparing SWRA with state-of-art ones prove that, SWRA is slightly more effective than GMM, but spares much less modeling time and memory. For the future work, we will engage to design an efficient method without float-precision operations. Such method will save the modeling time and satisfy requirements for hardware implementation.

ACKNOWLEDGMENT

This work is partially supported by grants from National Basic Research Program of China under contract No. 2009CB320906, the Chinese National Natural Science Foundation under contract No. 61035001, No. 61171139 and No. 61072095, and Fok Ying Dong Education Foundation under contract No. 122008.

6. REFERENCES

[1]C. Stauffer and W. E. L. Grimson, “Adaptive background mixture models for real-time tracking,” in *IEEE Conf. on Computer Vision and Pattern Recognition* (1999).

[2]D.-S. Lee, “Effective Gaussian mixture learning for video background subtraction,” in *IEEE T-PAMI* (2005).

[3]M. Haque, M. Murshed, and M. Paul, “Improved Gaussian mixtures for robust object detection by adaptive multi-background generation,” in *IEEE Conference on Computer Vision and Pattern Recognition* (2008).

[4]Y. Liu, H. Yao, W. Gao, et al., “Nonparametric background generation,” in *J. Vis. Commun. Image Represent* (2007).

[5]X. Zhang, L. Liang, Q. Huang, et al., “An efficient coding scheme for surveillance videos captured by stationary cameras,” in *Proc. Visual Commun. Image Process.* (2010).

[6]X. Zhang, L. Liang, T. Huang, et al., “A background model based method for transcoding surveillance videos captured by stationary camera,” in *PCS* (2010).

[7]M. Paul, W. Lin, C. T. Lau, et al., “MCFIS: Better I-frame for video coding,” in *IEEE Int. Conf. Control, Automatic, Robotics and Vision* (2010).

[8]M. Piccardi, “Background subtraction techniques: a review,” in *Proc. IEEE Man and Cybernetics* (2004).

[9]D. Liu, D. Zhao, X. Ji, et al., “Dual frame motion compensation with optimal long-term reference frame selection and bit allocation,” in *IEEE Trans. Circuits Syst. Video Technol.* (2009).

[10]T.K. Tan, G. Sullivan and T. Wedi, “Recommended simulation common conditions for coding efficiency experiments,” in *ITU-T Q.6/SG16*, VCEG-AA10, (2005)

[11]ftp://124.207.250.92/public/seqs/video