International Journal of Electrical and Computer Engineering (IJECE)
Vol. 8, No. 5, October 2018, pp. 2847~2856
ISSN: 2088-8708, DOI: 10.11591/ijece.v8i5.pp2847-2856

Background Estimation Using Principal Component Analysis Based on Limited Memory Block Krylov Subspace Optimization

Ilmiyati Sari[1], Asep Juarna[2], Suryadi Harmanto[3], Djati Kerami[4]
[1,2,3]Department of Computer Science and Information Technology, Gunadarma University, Indonesia
[4]Department of Mathematics, University of Indonesia, Indonesia

Article Info

Article history:

Received Nov 16, 2017
Revised Jan 12, 2018
Accepted Apr 1, 2018

Keyword:

Background estimation
Eigen vector
Krylov
PCA
Subspace optimization

ABSTRACT

Given a video of M frames of size $h \times w$. Background components of a video are the elements matrix which relative constant over M frames. In PCA (principal component analysis) method these elements are referred as "principal components". In video processing, background subtraction means excision of background component from the video. PCA method is used to get the background component. This method transforms 3 dimensions video ($h \times w \times M$) into 2 dimensions one ($N \times M$), where N is a linear array of size $h \times w$. The principal components are the dominant eigenvectors which are the basis of an eigenspace. The limited memory block Krylov subspace optimization then is proposed to improve performance the computation. Background estimation is obtained as the projection each input image (the first frame at each sequence image) onto space expanded principal component. The procedure was run for the standard dataset namely SBI (Scene Background Initialization) dataset consisting of 8 videos with interval resolution [146 150, 352 240], total frame [258,500]. The performances are shown with 8 metrics, especially (in average for 8 videos) percentage of error pixels (0.24%), the percentage of clustered error pixels (0.21%), multiscale structural similarity index (0.88 form maximum 1), and running time (61.68 seconds).

Corresponding Author:

Ilmiyati Sari,
Department of Computer Science and Information Technology,
Gunadarma University,
Margonda Raya Road, Pondok Cina, Depok, West Java 16424, Indonesia.
Email: ilmiyati.sari25@gmail.com

1. INTRODUCTION

Background subtraction is an important step in many computer vision systems to detect moving objects [1]. It is commonly used in video surveillance applications to detect persons, vehicles, animals, etc., before operating more complex processes for intrusion detection, tracking, people counting, etc. The basic operation consists of separating the moving objects called "foreground" from the static information called "background"[2]. It consists in using a model of the scene background in order to detect foreground objects by differencing incoming frames with the model. Indeed, the first step in background subtraction is background estimation.

We state the general problem of background estimation, also known as background initialization, bootstrapping, background reconstruction, initial background extraction, or background generation, as follows: Given a set of images of a scene taken at different times, in which the background is occluded by any number of foreground objects, the aim is to determine a model describing the scene background with no foreground objects [3].

Background Estimation based on eigenspace was firstly proposed by Oliver et al [4]. Foreground don't appear in the same location in the M sample images and they are typically small, do not have a significant contribution to modeling background. Consequently, the portions of an image containing a foreground cannot be well-described by this eigenspace model (except in very unusual cases), whereas the static portions of the image can be accurately described as a sum of the various eigenbasis vectors. That is, the eigenspace provides a robust model of the probability distribution function of the background, but not for the foreground.

The eigenspace model is formed by taking a sample of M images and computing both the mean and its covariance matrix. The eigenvectors of the covariance matrix are obtained by eigenvalue decomposition. In order to reduce the dimensionality of the space, in principal component analysis (PCA) [4]-[6] only k eigenvectors are kept, corresponding to the k largest eigenvalues. A CCTV (closed circuit television) can capture images in various resolutions, e.g., 176x120, or 352x240, or 704x480 pixels on the low to medium resolution range to 1280x720 or 1920x1080 pixels on the high resolution [7]. A CCTV with the lowest resolution, 176x120, size of the covariance matrix is 21120x21120. While computation eigenvalues and eigenvectors of large matrix take a long time [8].

The left singular vector at singular value decomposition of a normalized matrix which representation video was the principal component. Therefore, the singular value decomposition can yield the principal component without computing the covariance matrix. A subspace optimization technique to singular value decomposition significantly accelerates the classic simultaneous iteration method [9]. We propose Limited memory block Krylov subspace optimization for computing principal component. It will be used to construct background estimation.

2. RESEARCH METHOD

The following notation will be used: *italics* with subscripts to indicate vectors and matrix ($A_{h,w}$ is a matrix of h rows and w columns), bold letters with subscripts for images ($\boldsymbol{B}_{h,w}$ is an image with height h and width w). Given an image $\boldsymbol{I}$, of size h, w (height, width) it can be rearranged as a column vector $x_{N,1}$, where $N = h \cdot w$.

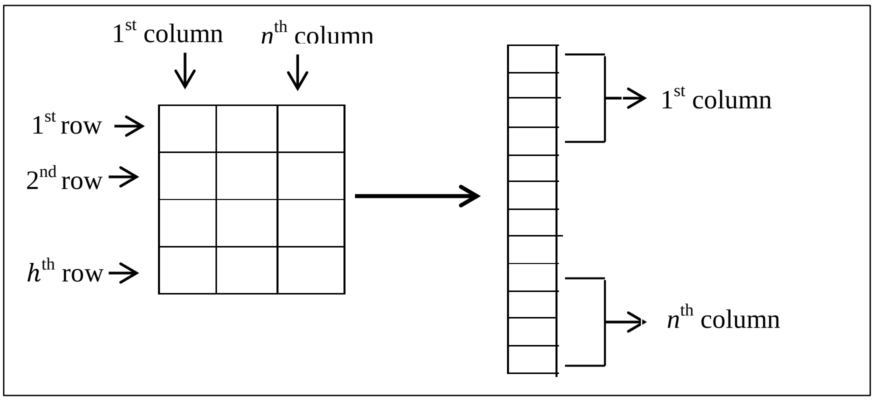


Figure 1. Conversion of $h \times w$ image into $N \times 1$ vector

Given a video consists of M frames $\boldsymbol{I}^1, \boldsymbol{I}^2, \cdots, \boldsymbol{I}^M$. These frames are arranged as a columns $x_{N,1}^1, x_{N,1}^2, \cdots, x_{N,1}^M$. These vectors $x_{N,1}^1, x_{N,1}^2, \cdots, x_{N,1}^M$ are formed into a matrix X with height N and width M,

$$X_{N,M} = [x_{N,1}^1 \quad x_{N,1}^2 \quad \cdots \quad x_{N,1}^M\,]$$

2.1. Principal Component via Limited Memory Krylov Subspace Optimization

Eigenvectors of covariance matrix can be obtained by singular value decompositon. Suppose normalized matrix which representation video, $A_{N,M} = \left(X_{N,M} - \mu \cdot I_{1,M}\right)$, where μ is average image and $I_{1,M}$ is row vector in which all elements are set to 1, singular value decomposition of matrix $A_{N,M}$ is

$$A_{N,M} = U_{N,M} \Sigma_{M,M} V_{M,N},$$

where $U_{N,M}$ is left singular matrix, $V_{M,N}$ is right singular matrix and $\Sigma_{M,M}$ is diagonal matrix in which diagonal elements are singular value. Multiplication $A_{N,M}$ with its transpose, $A^T_{M,N}$, is given by:

$$A_{N,M}A^T_{M,N} = U_{N,M}\Sigma_{M,M}V_{M,N}V^T_{N,M}\Sigma_{M,M}U^T_{M,N},$$
$$C_{N,N} = U_{N,M}\Sigma^2_{M,M}U^T_{M,N}. \quad (1)$$

Equation 1 is eigenvalue decomposition of covariance matrix, $C_{N,N}$, where columns of $U_{N,M}$ are eigenvectors of $C_{N,N}$. So, eigenvectors of covariance matriks are similar with left singular vector of $A_{N,M}$. Eigenvectors of covariance matrix corresponding to the most significant eigenvalues will yield the principal components. So, principal component can be produced by compute left singular vector of matrix $A_{N,M}$. X. Liu et all [9] proposed limited memory block krylov subspace optimization to compute dominant singular value decomposition. Its technique is proposed to significantly accelerate the simple subspace iteration method. The following we discuss how to obtain k largest singular value decomposition of matrix $A_{N,M}$ using limited memory block krylov subspace optimization. Starting from an initial point $X^{(0)} \in R^{N\times k}$, SSI (Simple Subspace iteration) computes the next iterate $X^{(i+1)}$ by the formula

$$X^{(i+1)} \in \text{orth}\left(AA^TX^{(i)}\right); \quad (2)$$

where orth (M) denotes the set of orthonormal bases for the range space of M. As such, the iterates of SSI, with a possible exception for the initial guess, satisfy the orthogonality condition $X^TX = I$. When k=1, the SSI method reduces to the well-known power method for computing the single largest eigenvalue and its eigenvector. In the SSI method, the orthonormalization step is indispensable (for example, see [10] for more details).

For an unstructured matrix A, the computational costs of the matrix-block multiplication (i.e., AA^TX) and orthonormalization in (2) are $O(MNk)$ and $O(Mk^2)$, respectively. In most applications, the approximating rank k is far less than the dimension m. Hence, the matrix-block multiplications of the type AA^TX constitute the dominate computational cost of SSI. Obviously, an acceleration will be achieved if one can reduce the number of iterations without having to incur extra matrix-block multiplications or other significant overhead. To achieve the goal of reducing the number of iterations, we propose to modify the basic SSI framework as follows. We replace the last iterate $X^{(i)}$ in the right-hand side of (2) by an "improved" intermediate iterate $\hat{X}^{(i)}$ so that

$$X^{(i+1)} \in \text{orth}\left(AA^T\hat{X}^{(i)}\right), \quad (3)$$

where, for a chosen subspace $\mathcal{S}^{(i)}$ with a block Krylov subspace structure,

$$\hat{X}^{(i)} := \arg\max_{X\in R^{N\times k}} \|A^TX\|^2_F, \text{ s.t.} X^TX = I,\ X \in \mathcal{S}^{(i)}. \quad (4)$$

Again, $X \in \mathcal{S}^{(i)}$ means all columns of X are from the subspace $\mathcal{S}^{(i)}$. The selection of the subspace $\mathcal{S}^{(i)}$ which is constructed from a limited memory of the last a few iterates. Its choice is of course not unique. We first consider the subspace spanned by the current i-th iterate and the previous p iterates; i.e.,

$$\mathcal{S}^{(i)} := span\left(X^{(i)}, X^{(i-1)}, \cdots, X^{(i-p)}\right); \quad (5)$$

where the *memory length* $p \geq 0$ will be specified in later. We collect the current and the other p saved iterate blocks in (5) into a matrix

$$\boldsymbol{X} = \boldsymbol{X}^{(i)}_p := \left[X^{(i)},\ X^{(i-1)}, \cdots, X^{(i-p)}\right] \in R^{N\times q}; \quad (6)$$

where $q = (p+1)k$ is the total number of columns in $\boldsymbol{X}^{(i)}_p$. For notational simplicity, from here on we often choose to drop the superscript and subscript from quantities like $\boldsymbol{X}^{(i)}_p$ whenever no confusion would arise. Also note that the collection matrix $\boldsymbol{X}$ is boldfaced to make it distinct from its blocks. Similarly, we define

$$\boldsymbol{Y} = \boldsymbol{Y}^{(i)}_p := \left[Y^{(i)},\ Y^{(i-1)}, \cdots, Y^{(i-p)}\right] \in R^{N\times q}, \quad (7)$$

which is also saved in memory. We emphasize that SSI already computes these blocks in $\boldsymbol{Y}$ and we only need to save them once computed. It is clearly that $X \in \mathcal{S}^{\{i\}}$ if and only if $X = \boldsymbol{X}V$ for some $V \in R^{q\times k}$, and the subspace optimization problem (4) is equivalent to a generalized eigenvalue decomposition problem:

$$\max_{V\in R^{q\times k}} \|(A^T X)V\|_F^2 \text{ s.t. } V^T(X^T X)V = I. \tag{8}$$

However, numerical difficulty may arise in solving (8) as the matrix $X^T X$ can become numerically rank deficient. A more stable approach, which we will implement, is to find an orthonormal basis for $\mathcal{S}^{(i)}$, say,

$$\boldsymbol{Q} = \boldsymbol{Q}_p^{(i)} \in \operatorname{orth}\left(X_p^{(i)}\right),$$

and to express a matrix $X \in \mathcal{S}^{(i)}$ as $\boldsymbol{X} = \boldsymbol{Q}V$ for some $V \in R^{q\times k}$. Here we assume that $\boldsymbol{X}$ has a full rank and will later relax this assumption. We now convert the generalized eigenvalue problem (8) into an equivalent eigenvalue problem

$$\max_{V\in R^{q\times k}} \|\boldsymbol{R}V\|_F^2 \text{ s.t. } V^T V = I, \tag{9}$$

where

$$\boldsymbol{R} = \boldsymbol{R}_p^{(i)} \coloneqq A^T \boldsymbol{Q}_p^{(i)}. \tag{10}$$

Next we describe how to calculate the matrix product $\boldsymbol{R}$ in (10) from historical information without any additional computation involving the matrix A. Since $\boldsymbol{Q} \in \operatorname{orth}(\boldsymbol{X})$and we assume that $\boldsymbol{X}$ has a full rank, there exists a nonsingular matrix $C \in R^{q\times q}$ such that $\boldsymbol{X} = \boldsymbol{Q}C$. Hence, $\boldsymbol{Q} = \boldsymbol{X}C^{-1}$, and $\boldsymbol{R}$ in (10) can be computed as

$$\boldsymbol{R} = A^T\boldsymbol{Q} = A^T\boldsymbol{X}C^{-1} = \boldsymbol{Y}C^{-1}, \tag{11}$$

Where $\boldsymbol{Y} = A^T\boldsymbol{X}$ is accessible from our limited memory. Once $\boldsymbol{R}$ is available, we can solve (9) by computing the k leading eigenvectors of the $q \times q$ matrix $\boldsymbol{R^T R}$. Let a solution to (9) be $\hat{V}$. The matrix product in equation (3) can then be assembled as

$$AA^T\hat{X}^{(i)} = A\boldsymbol{R}\hat{V} = A\boldsymbol{Y}C^{-1}\hat{V}. \tag{12}$$

The remaining issue is how to efficiently and stably compute $\boldsymbol{Q}$ and $\boldsymbol{R}$ even when the matrix X is numerically rank deficient. We use the following procedure in our implementation. Noting that each block in X is individually orthonormal, we choose to keep the latest block $X^{(i)}$ intact, and project the rest of the blocks onto the null space of ${X^{(i)}}^T$, obtaining

$$\boldsymbol{P}_X = P_X^{(i)} \coloneqq \left(I - X^{(i)}\left(X^{(i)}\right)^T\right)\left[X^{(i-1)}\ X^{(i-2)}\ \cdots\ X^{(i-p)}\right]. \tag{13}$$

Next we perform an orthonormalization of $\boldsymbol{P}_X$ via the eigenvalue decomposition of its Gram matrix

$$\boldsymbol{P}_X^T\boldsymbol{P}_X = U_X\Lambda_X U_X^T. \tag{14}$$

where U_X is orthogonal and Λ_X is diagonal. It can be easily verified that the matrix

$$\boldsymbol{Q} = \boldsymbol{Q}_p^{(i)} \coloneqq \left[X^{(i)}, \boldsymbol{P}_X U_X \Lambda_X^{-\frac{1}{2}}\right] \in orth\ \boldsymbol{P}_X^{(i)}, \tag{15}$$

provided that Λ_X is invertible. The above procedure can be stabilized by monitoring the numerical rank of $\boldsymbol{P}_X$, or specifically the eigenvalues on the diagonal of the matrixΛ_X in (14). We choose to implement the following two-step stabilization scheme:

Step 1 Delete the columns of $\boldsymbol{P}_X$ whose Euclidean norms are below a threshold $\epsilon_1 > 0$.

Step 2 Delete the eigenvalues in Λ_X, and corresponding columns in U_X, that are less than $\epsilon_2 > 0$.

With a slight abuse of notation, we will continue to use $\boldsymbol{P}_X$, U_X and Λ_X to denote their stabilized versions, respectively, after possible deletions. Therefore, a stable construction of $\boldsymbol{Q}$ is still given by formula (15). After this stable orthonormalization, the corresponding $\boldsymbol{R}$ matrix can be generated as

$$\boldsymbol{R} = \boldsymbol{R}_p^{(i)} := \left[Y^{(i)}, \boldsymbol{P}_Y U_X \Lambda_X^{-\frac{1}{2}}\right], \tag{16}$$

where before the stabilization procedure we had

$$\boldsymbol{P}_Y = P_Y^{(i)} := \left[Y^{(i-1)} \ \cdots \ Y^{(i-p)}\right] - Y^{(i)}\left(X^{(i)}\right)^T\left[X^{(i-1)} \ \cdots \ X^{(i-p)}\right], \tag{17}$$

but some of the columns of $\boldsymbol{P}_Y$ may have been deleted corresponding to those deleted columns of $\boldsymbol{P}_X$ during the stabilization steps. After the removal of numerical rank deficiency, the $\boldsymbol{R}$ matrix in (16) is well defined as is the $\boldsymbol{Q}$ matrix in (15).

In summary, the algorithm performs eigenvalue decompositions on two small symmetric positive definite matrices: $\boldsymbol{P}_X^T\boldsymbol{P}_X$ in (14) and $\boldsymbol{R}^{\boldsymbol{T}}\boldsymbol{R}$ in (9). The sizes of the two matrices are pk and $(p+1)k$, respectively, and frequently smaller due to deletions. Our computational experience indicates that in general p should be set to 2 or 3, or at most 4 but not greater. Consequently, when k is sufficiently smaller than N, it holds that $(p + 1)k \ll M < N$.

2.1.1. Memory Length

The memory length p, used for constructing the subspace $\mathcal{S}^{(i)}$ in (5), is a crucial parameter to the performance of our algorithm. The simplest way is to assign a constant integer value p_{max} to p at every iteration once the iteration counter i reaches p_{max}; that is, at iteration i,

$$p = min(i, p_{max}). \tag{18}$$

In general a larger p_{max} leads to a smaller number of iterations, but increasing p_{max} also increases the computational costs per iteration. Our computational experiments indicate that usually a good balance is attained for $p_{max} \in \{2, 3, 4\}$.

We have also found that an adaptive strategy on selecting p is useful to improving the performance of LMSVD. As the iterate sequence converges, the neighboring iterates tend to become more and more linearly dependent. Therefore, once judged appropriate it is beneficial to shrink the memory by deleting a block from the memory, reducing the size of later subspace optimization problems. Specifically, after p_{max} iterations, we activate the following adaptive memory size strategy:

$$p = \left\lceil \frac{N_{C(\boldsymbol{R})}}{k} \right\rceil - 1, \tag{19}$$

where $\lceil t \rceil$ is the smallest integer greater than or equal to t, and $N_{C(R)}$ is the number of columns in $\boldsymbol{R}$ which can be smaller than$(p+1)k$ due to possible deletions done in the two stabilization steps. Combining (18) and (19), we reach our formula for selecting the memory length p at the i-th iteration:

$$p = min\left\{i, \left\lceil \frac{N_{C(\boldsymbol{R})}}{k} \right\rceil - 1, p_{max}\right\}, \tag{20}$$

which is nonnegative. Generally, p initially increases to reach p_{max}, then becomes non-increasing with a probability to decrease to a smaller value, even possibly to zero. Of course, when the memory length p becomes zero, our method reduces to the classic SSI.

2.1.2. LMSVD Algorithm

Based on the description above, we state our full Algorithm. For ease of reference, the algorithm will be referred to as LMSVD.

```
Algorithm LMSVD: Limited Memory Block Subspace Krylov Optimization for SVD
Input $A_{N,M}$ and $k$. Initialize $\boldsymbol{X} = X^{(0)} \in R^{m\times k}$, $\boldsymbol{Y} = Y^{(0)} = A^T X^{(0)}$, and $p = i = 0$;
while “not converged” do
    /* Block Subspace Optimization */
    Compute $\boldsymbol{P}_X$ by (13) and perform stabilization Step 1;
    Compute $\boldsymbol{P}_Y$ by (17) with the same column deletions as for $\boldsymbol{P}_X$;
    Compute the eigenvalue decomposition of $\boldsymbol{P}_X^T \boldsymbol{P}_X$ in (14);
    Perform stabilization Step 2 to possibly shrink $\Sigma_X$ and $U_X$;
    Compute $\boldsymbol{R}$ by (16) and eigenvalue decomposition of $\boldsymbol{R}^T \boldsymbol{R}$;
    Let $\hat{V}$ solve (9), consisting of the $k$ leading eigenvectors of $\boldsymbol{R}^T \boldsymbol{R}$;
    Compute $\hat{X}^{(i)} = \boldsymbol{Q}\hat{V}$ and $\hat{Y}^{(i)} = \boldsymbol{R}\hat{V}$ (which equals $A^T \hat{X}^{(i)}$);
    /* Simultaneous Iteration */
    Compute $X^{(i+1)} \in \mathrm{orth}\left(A\hat{Y}^{(i)}\right)$ and $Y^{(i+1)} = A^T X^{(i+1)}$;
    Increment $i$, update $p$, $\boldsymbol{X}$ and $\boldsymbol{Y}$, and continue.
Output $U, \Sigma$, and $V$
```

2.2. Background Estimation

Once principal component (U) and the mean background (μ) are computed, the input image (γ_i) with foreground objects was subtracted by the mean background. Defining an principal component matrix as $U = [u_1, u_2, \cdots, u_k]$. It follows that the coordinate (weight) in eigenspace of input image, γ_i, can be computed as follows

$$\omega = (\gamma_i - \mu)^T U, \tag{21}$$

when ω is back projected onto the image space, a background estimation is created

$$\gamma_R = U\omega^T + \mu. \tag{22}$$

Noting that since the principal component matrix describes the general background appearances well however not the small moving objects, γ_R does not contain small objects.

2.3. Metrics

The metrics adopted to evaluate the accuracy of the estimated background models have been chosen among those used in the literature for background estimation [3]. Denoting with GT (Ground Truth) an image containing the true background and with CB (Computed Background) the estimated background image computed with one of the background initialization methods, the eight adopted metrics are:

a. Average Gray-level Error (AGE): It is the average of the gray-level absolute difference between GT and CB images. Its values range in [0, L-1], where L is the maximum number of grey levels; the lower the AGE value, the better is the background estimate.
b. Total number of Error Pixels (EPs): An error pixel is a pixel of CB whose value differs from the value of the corresponding pixel in GT by more than some threshold τ (in the experiments the suggested value τ=20 has been adopted). EPs assume values in [0; N], where N is the number of image pixels; the lower the EPs value, the better is the background estimate.
c. Percentage of Error Pixels (pEPs): It is the ratio between the EPs and the number N of image pixels. Its values range in [0, 1]; the lower the pEPs value, the better is the background estimate.
d. Total number of Clustered Error Pixels (CEPs): A clustered error pixel is defined as any error pixel whose 4-connected neighbors are also error pixels. CEPs values range in [0, N]; the lower the CEPs value, the better is the background estimate.
e. Percentage of Clustered Error Pixels (pCEPs): It is the ratio between the CEPs and the number N of image pixels. Its values range in [0,1]; the lower the pCEPs value, the better is the background estimate.
f. Peak-Signal-to-Noise-Ratio (PSNR): It is defined as $PNSR = 10 \cdot \log_{10}((L-1)^2/MSE)$; where L is the maximum number of grey levels and MSE is the Mean Squared Error between GT and CB images. This frequently adopted metric assumes values in decibels (db); the higher the PSNR value, the better is the background estimate.
g. MultiScale Structural Similarity Index (MS-SSIM): This is the metric defined in [11], that uses structural distortion as an estimate of the perceived visual distortion. It assumes values in [0; 1]; the higher the value of MS-SSIM, the better is the estimated background.

h. Color image Quality Measure (CQM): It is a recently proposed metric [12], based on a reversible transformation of the YUV color space and on the PSNR computed in the single YUV bands. It assumes values in db and the higher the CQM value, the better is the background estimate.

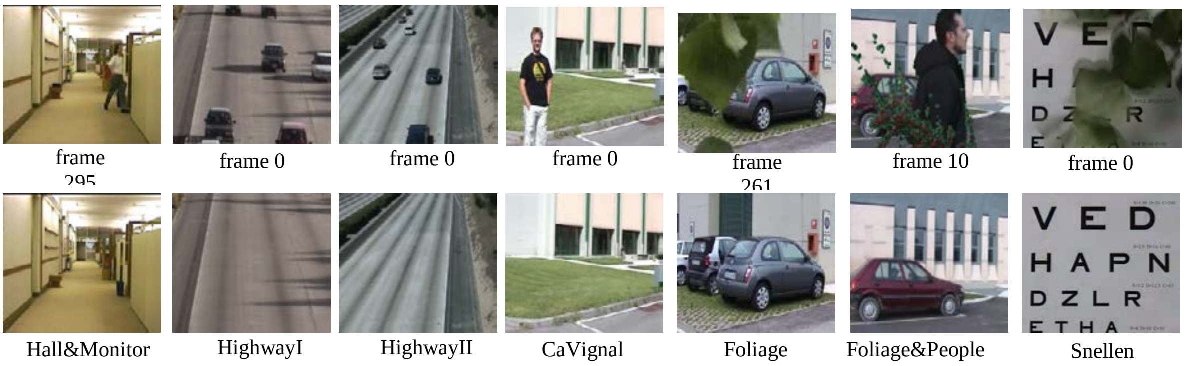
While the last metric is defined only for color images, metrics 1 through 7 are expressly defined for gray-scale images. In the case of color images, they are generally applied to either the gray-scale converted image or the luminance component Y of a color space such as YcbCr .

3. RESULTS AND ANALYSIS

This section describes the results of the implementation of the proposed method. The result is background estimation using principal component. The metrics adopted to evaluate the accuracy of the estimated background models have been chosen among those used in the literature for background estimation.

3.1. Background Estimation

In this paper, the Scene Background Initialization (SBI) data set was chosen for the background estimation. The data set contains seven image sequences and corresponding ground truth (GT) backgrounds are given in Figure 1. In Table 1 we report, for each sequence, the name, the number of available frames, the subset of the frames adopted for testing, and the original resolution. The subsets have been selected in order to avoid the inclusion into the testing sequences of empty frames (frames not including foreground objects). The ground truths (GT) have been manually obtained by either choosing one of the sequence frames free of foreground objects (not included into the subsets of used frames) or by stitching together empty background regions from different sequence frames. Both the complete SBI dataset and the ground truth reference background images were made available through the SBMI 2015 website at http://sbmi2015.na.icar.cnr.it [3].


frame 295 frame 0 frame 0 frame 0 frame 261 frame 10 frame 0

Hall&Monitor HighwayI HighwayII CaVignal Foliage Foliage&People Snellen

Figure 1. Example frames from the seven sequences of the SBI dataset (first row) and coressponding GT (second row)

Table 1. Information on Sequences Adopted for Evaluation

Name	Original frames	Used frames	Resolution
Hall&Monitor	0-299	4-299	352×240
HighwayI	0-439	0-439	320×240
HighwayII	0-499	0-499	320×240
CaVignal	0-257	0-257	200×136
Foliage	0-399	6-399	200×148
People&Foliage	0-349	0-340	320×240
Snellen	0-333	0-320	146×150

This algorithm is implemented on Matlab R2015a at personal computer with intel core i-5 processor and 4 GB RAM. the result of the implementation is the background estimation using the principal component via the LMSVD algorithm. The frame in the first row of Figure 1 is used as γ_i in equation 25, so we get the background estimation in Figure 2 using equation (26). The performance of the proposed method is evaluated both qualitative and quantitative. In Figure 2 we show the background images obtained by the proposed methods on the SBI dataset, while in Table 2 we report accuracy results according to the metrics described in 2.3.

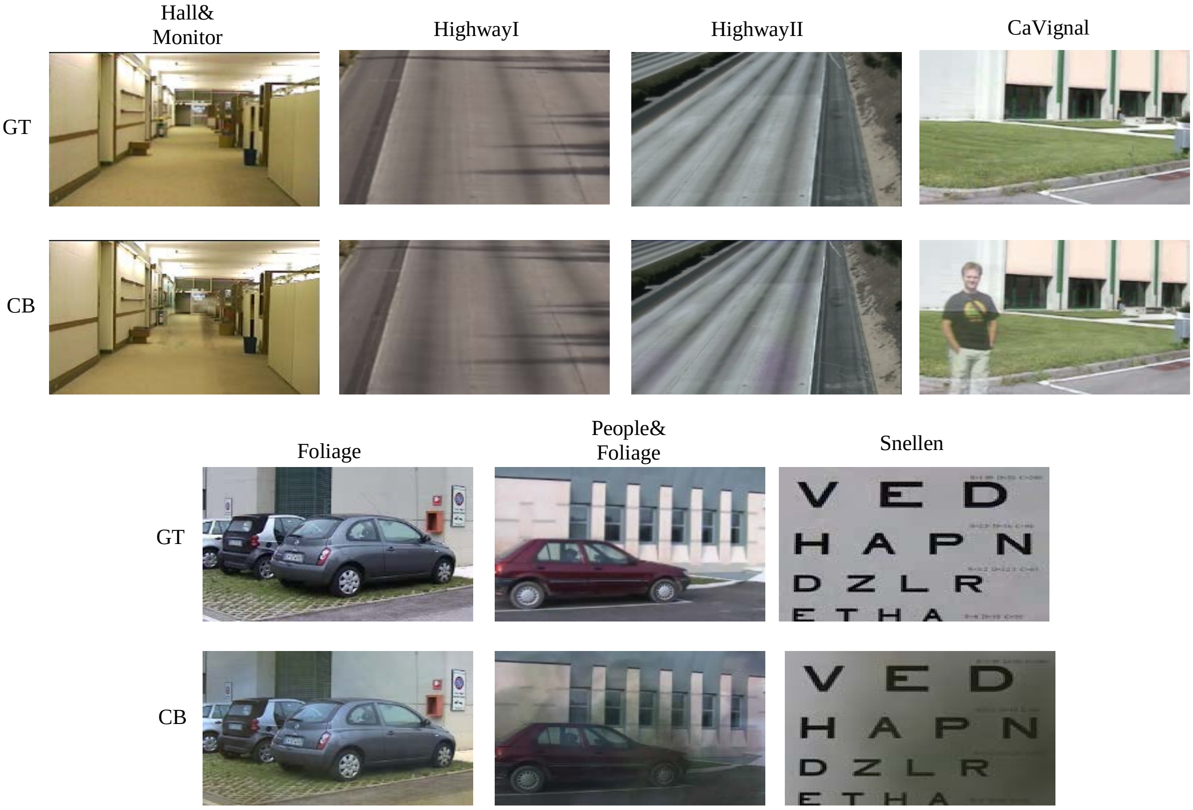


Figure 2. Background estimation, GT (Ground Truth) and CB (Computed Backgrounds were obtained by proposed method)

Table 2. Accuracy Results of the Proposed Methods on the SBI Dataset

Sequence	AGE	EPs	pEPs	CEPs	pCEPs	MSSSIM	PSNR	CQM
HighwayII	3,24	273	0,00	1	0,00	0,98	33,24	32,92
HighwayI	3,71	2571	0,03	1400	0,04	0,95	29,82	29,82
Foliage	3,87	1117	0,01	382	0,03	0,96	28,97	28,91
Hall&Monitor	8,77	2849	0,10	2241	0,08	0,83	21,67	21,65
CaVignal	12,14	4638	0,16	2778	0,09	0,95	24,32	23,74
PeopleAndFoliage	34,93	47706	0,62	41913	0,54	0,74	15,24	14,76
Snellen	43,55	15771	0,76	14416	0,69	0,75	14,25	14,16
Average	15,74	10704	0,24	9019	0,21	0,88	23,93	23,71

For sequence Hall&Monitor, a man walking straight down the corridor occupies the same image region for more than 65% of the sequence frames, while the briefcase is left on the small table for the last 60% of sequence frames. The proposed method well handle the abandoned briefcase, but it includes a man who walks a bit obscenely. For both HighwayI and HighwayII sequences, the proposed method succeed in providing a faithful representation of the background model. This is due to the fact that, even though the highway is always fairly crowded by passing cars, the background is revealed for at least 50% of the entire bootstrap sequence length and no cars remain stationary during the sequence.

For sequence CaVignal, the only man appearing in the sequence stands still on the left of the scene for the first 60% of sequence frames; then starts walking and rests on the right of the scene for the last 10% of sequence frames. The persistent clutter at the beginning of the scene leads of the proposed methods to include the man on the left into the estimated background. For sequence Foliage, even though moving leaves occupy most of the background area for most of the time, the proposed methods achieve a quite good representation of the scene background.

For sequence People&Foliage, the artificially added leaves and men occupy almost all the scene area in almost all the sequence frames. The proposed methods to include the contribution of leaves and men into the final background model. In sequence Snellen, the foreground leaves occupy almost all the scene area in almost all the sequence frames. The proposed methods achieve a quite good representation of the scene background. However, the scene background have low luminance than GT.

In order to assess the challenge that each sequence poses for the proposed methods, we computed all metrics obtained by the proposed methods for each sequence, and ranked the sequences according to these metrics, as shown in Table 2. Here, HighwayI and HighwayII sequences reveal as those that are best handled, while Snellen is the worst handled. Bearing in mind the kind of foreground objects included into the sequences, we can observe that their size is not a major burden; e.g., Foliage sequence is better handled than Hall&Monitor, even though the size of the foreground objects is much larger. As instance, CaVignal sequence is worse handled than Foliage, since it includes almost static foreground objects that are frequently misinterpreted as background. It can also be observed that the values of pEPs and MS-SSIM metrics perfectly vary according to the difficulty in handling the sequences; these two metrics confirm to be strongly indicative of the performance of background initialization methods.

Conventional PCA can not be applied to get background estimation for the SBI dataset because the covariance matrix size of all sequences is too large as shown in Table 3. The hardware memory used is not sufficient to obtain the eigenvector of the covariance matrix (principal component). Table 3 shows the performance of the proposed method. The method succeed to produce the principal component for all sequence of images with time varies depending on total frames of sequences.

Table 3. Performance of Proposed Method

Sequence	Resolution	Size of covariance matrix	Total frames	time (proposed method)
Hall&Monitor	352×240	84480×84480	295 frames	39,84
HighwayI	320×240	76800×76800	440 frames	95,95
HighwayII	320×240	76800×76800	500 frames	223,83
CaVignal	200×136	27200×27200	258 frames	6,15
Foliage	200×148	29600×29600	394 frames	8,72
People&Foliage	320×240	76800×76800	341 frames	54,87
Snellen	146×150	21900×21900	321 frames	2,37
Average				61,68
Sequence	Resolution	Size of covariance matrix	Total frames	time (proposed method)

4. CONCLUSION

In this paper, we have presented a background estimation using the principal component. The principal component is obtained by compute the dominant singular value decomposition using the limited memory Krylov subspace optimization. The columns of left singular matrix of the dominant singular value decomposition is the principal component. Background estimation is obtained as the projection each input image (the first frame at each sequence image) onto space expanded principal component.

The procedure was run for the standard dataset namely SBI (Scene Background Initialization) dataset consisting of 8 videos with interval resolution [146 150, 352 240], total frame [258,500]. The performances are shown with 8 metrics, especially (in average for 8 videos) percentage of error pixels (0.24%), the percentage of clustered error pixels (0.21%), multiscale structural similarity index (0.88 form maximum 1), and running time (61.68 seconds).

REFERENCES

[1] A. Sobral, *et al.*, “Comparison of Matrix Completion Algorithms for Background Initialization in Videos,” *Springer International Publishing*, pp. 510-517, 2015.

[2] A. Sobral and A. Vacavant, “A Comprehensive Review of Background Subtraction Algorithms Evaluated with Synthetic and Real Videos,” *Elsevier Journal*, pp. 4-21, 2014.

[3] L. Maddalena and A. Petrosino, “Towards Benchmarking Scene Background Initialization,” *Springer International Publishing Switzerland*, pp. 469-476, 2015.

[4] N. M. Oliver, *et al.*, “A Bayesian Computer Vision System for Modeling Human Interactions,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol/issue: 22(7), pp. 831-843D, 2000.

[5] H. Sabrol and S. Kumar, “Recognition of Tomato Late Blight by Using DWT and Component Analysis,” *International Journal of Electrical and Compuer Engineering (IJECE)*, vol/issue: 7(1), pp. 194-199.

[6] Venkatramaphanikumar S. and K. V. K. Kishore, “Face Recognition with Modular Two Dimensional PCA under Uncontrolled Illumination Vaiations,” *International Journal of Electrical and Compuer Engineering (IJECE)*, vol/issue: 6(4), pp. 1610-1616.

[7] www.cctvcamerapros.com

[8] M. Turk and A. Pentland, “Eigenfaces for recognition,” *Journal of Cognitive Neuroscience*, vol/issue: 3(1), pp. 71-86, 1991.

[9] X. Liu, *et al.*, “Limited Memory Block Krylov Subspace Optimization for Computing Dominant Singular Value Decomposition,” *SIAM Journal on Scientific Computing*, vol/issue: 35(3), pp. A1641-A1668, 2013.

[10] Y. Saad, “Numerical Methods for Large Eigenvalue Problems,” Manchester University Press, 1992.

[11] Z. Wang, *et al.*, “Multiscale structural similarity for image quality assessment,” in *Signals, Systems and Computers, 2004. Conference Record of the Thirty-Seventh Asilomar Conference on*, vol. 2, pp. 1398-1402, 2003.

[12] Y. Yalman and I. Erturk, “A new color image quality measure based on YUV transformation and PSNR for human vision system,” *Turkish J. Of Electrical Eng. & Comput. Sci.*, vol. 21, pp. 603-612, 2013.

BIOGRAPHIES OF AUTHORS

Ilmiyati Sari received her B.S. degree in Mathematics from University of Indonesia in 2009, her M.S. in Mathematics from University of Indonesia in 2012 and. She has published extensively in the area of video processing. Her research interests include dynamic models, statistics, image processing, and video processing.
Email:ilmiyati@staff.gunadarma.ac.id

Asep Juarna received his doctoral degree in Theoretical Informatics from Universite de Bourgogne, France. Currently, he is an associate professor at the Department of Informatics of Gunadarma University. His research interest include numerical analysis, modeling and simulation, and combinatorics. His current researches are applied combinatorics in portfolio of stock, video processing, and data clustering modeling.
Email:ajuarna@staff.gunadarma.ac.id

Suryadi Harmanto received the master degree in information technology from Gunadarma University. Currently, he is a professor at the Gunadarma University. His research interest on computational mathematics and information technology.
Email:suryadi_hs@staff.gunadarma.ac.id

Djati Kerami received the Ph.D degree in Informatics from Institut National Polytechnique de Toulouse, France. Currently, he is a professor at the Dept. of Mathematics of University of Indonesia.His research interest include computational mathematics, combinatorial algorithms, and bioinformatics.
Email:djatikr@sci.ui.ac.id