CSE6443 Numerical Linear Algebra Final Project Report

Real-time background/foreground separation of videos using matrix decomposition

Fangzhou Liu, PhD 7th CEE, Nimisha Roy, PhD 2nd CEE, Asra Yousuf, MS 1st CSE

Abstract

Change detections and background removal from a video sequence form a fundamental and critical task for many computer-vision applications. Various challenges still present in the current algorithms for background subtraction, since such approaches need to be robust to the changes of illumination and non-stationary background objects (e.g. snow, rain, and leaves), and to be sensitive to the movement of the target objects. In this study, we compared the performance of four background subtraction algorithms, including single value decomposition (SVD), randomized single value decomposition (rSVD), non-negative matrix factorization (NMF), and robust principle component analysis (rPCA), for detecting moving vehicles and pedestrians in both static and dynamic background settings. The performance of the algorithms is measured by accuracy, precision, recall, and f-measures. The results have shown that rPCA yields the best performance with some marginal improvements as compared to the other algorithms.

Introduction

Background separation is a major preprocessing step in many vision-based applications, especially used in motion detection, identification, tracking, activity analysis and semantic description of objects of interest in a relatively more static background. The main aim of background subtraction is to separate moving object foreground from the background in a video, which always makes the subsequent video processing tasks easier and more efficient. For example, while studying cars moving on a freeway, the unnecessary elements like trees, building, street lights etc. can be eliminated so that only the main object of concern i.e., the cars, remain in the video. Our proposal is to demonstrate the application of matrix decomposition techniques to the problem of background separation. We aim to separate background from videos using four matrix decomposition techniques, viz, Singular Value Decomposition (SVD), randomized singular value decomposition (rSVD), Non-negative matrix factorization (NMF) and Robust Principal Component Analysis (rPCA) and compare the performance of the four methods based on speed, precision, recall, f-score and accuracy (sensitivity to moving background, illumination changes, shadows, etc.).

Literature Review

Background separation has been used as a crucial component in many fields of computer vision applications, such as video surveillance to detect human activity, human-computer interface, motion detection and multimedia applications (Sobral et al., 2016)

A common assumption for background removal is that certain attributes of the objects of interest (foreground) change rapidly and should be analyzed in a different manner from objects changing slowly (background). However, deciding which attributes are the correct ones and how to measure change can be subtle. Some of the work in literature focusing on background removal use methods like Gaussian mixture models for detecting static foreground regions of images (Tian et al, 2005), Grassmannian Robust Adaptive Subspace Tracking based on principal component analysis applied to estimated subspace of dataset (He et al., 2012), Dynamic mode decomposition based on principal component analysis applied to Fourier

decomposition of video frames (Grosek & Kutz,2014), Incremental Non-negative matrix factorization (Bucak et al., 2007) and Local SVD Binary Pattern feature descriptor based on singular value decomposition (Guo et al., 2016).

Computational cost and real-time processing are of utmost importance in deciding the methodology to use for background detection. For instance, there are an estimated minimum 10,000 surveillance cameras in the city of Chicago (Babwin, 2010), and these video datasets are almost entirely used post-hoc for analysis. The goal of background removal is to enable technologies that can analyze video data in real-time with minimum computational cost.

Problem Statement & Applications

The problem statement is to apply matrix decomposition techniques to separate the moving foreground of a video from its stationary background and to evaluate and compare the performance of these techniques on different datasets with different application environments.

Simple motion detection algorithms compare a static background frame with the current frame of a video scene, pixel by pixel which becomes the basic principle of background subtraction; and it serves as a critical component in the applications of automatic video surveillance ranging from counter-terrorism to home security to pet monitoring. The evaluations of background subtraction methods with respect to the challenges of video surveillance suffer from various short-comings, and this study proposes to evaluate the performance of some of the basic methods of background removal in such multi-media applications. The goal is to better enable us to provide a general guide on method selections for various techniques under different ground truth scenarios.

Algorithms and Methods

To facilitate the process of background removal of an input video, the first step is to split the video into frames based on a time frame resolution. Each image frame is then converted into a one-dimensional vector. These vectors are stacked horizontally to form a matrix (A) where each column of the matrix corresponds to each image frame of the video. Matrix decomposition techniques are then used to approximate the matrix as a low rank matrix component (corresponding to the background) and a sparse matrix component (corresponding to the foreground). For the purposes of background removal, the value of k is taken as 2 in this project. The matrix decomposition can be achieved using many techniques. Some of them that are implemented in this project are as follows:

- **Singular Value Decomposition (SVD)**

  SVD provides a means to perform factorization for a complex matrix, with a wide range of applications in signal process and statistics, which can be expressed as follows:

  $$M = U\Sigma V^T$$

  where matrix M is a $m \times n$ matrix, U ($m \times n$) and V ($n \times n$) are orthogonal matrices, respectively, and Σ (n×n) is a diagonal matrix. The SVD is a generalization of the Eigen decomposition which can be used to analyze rectangular matrices; it can be applied to separate the background of the 1-D vectors of the stacked image by identifying the 'non-changing' parts of the matrix (i.e. the background) and through decomposition to obtain the first k singular values of Σ. This method has a high computational expense. This study constructed a simple SVD code in lieu of applying the built-in function of MATLAB; the idea is to use the QR decomposition on the target matrix A to

iterate the *U* out from the left and then use QR on *A* transposed to yield *V* out from the right. The iteration transforms *A* lower triangular and then upper triangular alternately, until it becomes both upper and lower triangular at the same time, with singular values on the diagonal. The Householder method is used for QR decomposition. The algorithm for SVD is shown in Figure 1.

Algorithm 1: Matrix decomposition using singular value decomposition

Input: A, k
Output: U, S, V
$U = I_m$, $S = A^T$, $V = I_n$
while S not diagonalized **do**
  $[\mathbf{Q}, \mathbf{S}] = \text{House\_QR}(\mathbf{S^T})$;
  $[\mathbf{Q}, \mathbf{S}] = \text{House\_QR}(\mathbf{S^T})$;
  If S becomes diagonal, **break**;
end
function $[\mathbf{A}, \mathbf{b}] = \text{House\_QR}(P)$
for $k = 1{:}n$ **do**
  determine $\hat{P}_k$ of order $m - k + 1$ so that ;

$$\hat{P}_k \begin{bmatrix} a_{k,k} \\ \vdots \\ a_{m-1,k} \\ a_{m,k} \end{bmatrix} = \begin{bmatrix} r_{k,k} \\ 0 \\ \vdots \\ 0 \end{bmatrix};$$

  $A = diag(I_{k-1,\hat{p}_k}, A)$;
  $b = diag(I_{k-1,\hat{p}_k}, b)$;
end

Figure 1: Algorithms of Singular value decomposition

- **Randomized Singular Value Decomposition (RSVD)**

The size of the matrix constructed from the stacked frame of our dataset is of the order of about 10^8 elements, which would cause significant demand on the memory of any workstations, and result in memory runout. To handle such size of matrix, this study followed the suggestion by Halko et al. (2009) and re-implemented the method by adding the randomized algorithm to construct the low-rank approximate matrix factorizations to reduce the dimension of the original matrix. With a matrix *Q*, rSVD only need to perform SVD on *Q*A*, where $A \approx Q(Q^*A)$, to generate the $U'\Sigma V^T$ where $U=QU'$. By using such an approximation approach, the size and rank of the matrix for SVD is significantly reduced to save computational power. To find matrix *Q*, this study draws a random test matrix Ω to form the matrix product $Y = A\Omega$, and in such way the QR factorization also became computationally less challenging. The algorithm of RSVD is shown in Figure 2.

Algorithm 2: Matrix decomposition using randomized singular value decomposition

Input: A, k
Output: U, S, V
Initialize $\boldsymbol{\Omega}$ as a random matrix with each entry from the standard normal distribution.
$Y = A\Omega$
$[Q, R] = \text{MGS\_QR}(Y)$ % QR decomposition using Modified Gram Schmidt
$[U', S, V] = \text{SVD}(Q^T A)$ % SVD using Algorithm 1
$U = QU$

Figure 2: Algorithms of Randomized Singular value decomposition

- **Non-negative Matrix Factorization (NMF)**

  Non-negative matrix factorization approximates a non-negative matrix A by a product of two non-negative low rank factor matrices W and H, as follows:

  $$A = W.H$$

  where the columns of W are basis vectors and columns of H are in one-to-one correspondence with the basis vectors of W. W is the low rank approximated matrix chosen under the constraint that W and H are non –negative matrices. Classical NMF methods minimize either the Euclidean distance or the Kullback-Leibler divergence between A and W^TH. In this project, the objective function to be minimized is defined as the root mean squared residual between A and W^TH. The factorization uses an iterative method starting with random initial values for W and H based on the multiplicative update formula introduced by Lee and Seung (2001). Because the objective function often has local minima, repeated factorizations yield different W and H values. Hence, the thresholding is based on maximum iterations (taken as 100) and minimum tolerance on the residual (taken as 1e -4). The algorithm for NMF is shown in Figure 3.

Algorithm 3: Matrix decomposition using Non-negative matrix factorization

Input : A, k
Output: W, H
$[m, n] = size(A)$
$maxiter = 100; tolx = 1e-4$
$W = rand(m, k); H = rand(k, n)$
Procedure NNM_factor (A, k)
for $j = 1$ *to* $maxiter$ **do**
  $H = H * (WTA)./(WTWH + 10^(-9))$
  $W = W * (AHT)./(WHHT + 10^(-9))$
  $d\_norm = 0.5 * ||A - WH||2$
  if $d\_norm < tolx || j == maxiter$ **then**
    break
  end
end

Figure 3: Non- negative matrix factorization algorithm

- **Robust Principal Component Analysis (RPCA)**

  PCA is one of the most widely used statistical tool for data analysis and dimensionality reduction. The low-rank component L_0 of the matrix naturally corresponds to the stationary background and the sparse component S_0 captures the moving objects in the foreground. However, given the size of each matrix M, it is extremely challenging to derive L_0 using the classical approach. Robust PCA is a modification made on top of the classical PCA, especially applicable for datasets populated with grossly corrupted observations- a phenomena fairly common in problems of image processing, web data analysis, and bioinformatics, where some measurements may be arbitrarily corrupted (due to occlusions, malicious tampering, or sensor failures) or simply irrelevant to the low-dimensional structure we seek to identify. RPCA aims to solve this problem by recovering a low-rank matrix L_0 from highly corrupted measurements.

  $$M = L_0 + S_0.$$

Algorithm 4: Matrix decomposition using robustPCA(via Primary Component Pursuit (PCP)

Input : A, k
Output: L, S
$[m, n] = size(A)$
$maxiter = 10; tol = e^{-9}$
$S_0 = 0; L_0 = 0; Y_0 = 0, \mu > 0$
Procedure PCP (A, k)
while *not converged* **do**
  $Compute: L_{k+1} = D_\mu(A - S_k - \mu^{-1}Y_k)$
  $Compute: S_{k+1} = S_{\lambda\mu}(A - L_{k+1} - \mu^{-1}Y_k)$
  $Compute: Y_{k+1} = Y_k + \mu(A - L_{k+1} - S_{k+1})$
end

Figure 4: Robust PCA using Primary Component Pursuit algorithm

Experimental Setting/Tests/Data Sets

The datasets used in the study are open source image frame sets with the corresponding ground truth image frames that are readily accessible for evaluation at http://changedetection.net/ (Wang et al., 2014). We implemented the four matrix decomposition techniques on three distinct datasets, to account for different challenges encountered in background modelling owing to complex application environments. More specifically, the three datasets are explained below:

a) Static background condition (Case-A): This was the baseline case scenario with static background of a highway and moving vehicles as the foreground objects. 700 image frames were used for analysis with the size of an image being 380*420 pixels, resulting in the use of a compression factor of 0.5 for matrix A (to avoid memory runout).
b) Video noise/Camera Jitter condition (Case B): The datasets used in this case scenario was that of two sportsperson playing badminton, with the badminton court being the static background and the players being the moving foreground objects. There was also another dataset of people walking in a train station. 500 image frames were used for the analysis of both datasets with the size of an image being 720*480 pixels, resulting in the use of a compression factor of 0.25 for matrix A.
c) Dynamic background condition (Case-C): The dataset used in this case scenario was that of a canoe sailing in the sea, with the canoe being the moving foreground object and the continuously moving water as the dynamic background. 700 image frames were used for the analysis with the size of an image being 320*240 pixels, resulting in the use of a compression factor of 0.5 for matrix A.

The four algorithms were implemented on the three datasets and their performance was measured based on four parameters:

$$precision = \frac{tp}{tp + fp}, recall = \frac{tp}{tp + fn}, \quad f - measure = \frac{2 \times precision \times recall}{precision + recall}$$

$$accuracy = \frac{tp+tn}{tp+tn+fp+fn}$$

where positive is regarded as foreground and negative is regarded as background. So, *tp* gives the number of correctly detected foreground pixels, *tn* gives the number of correctly identified background pixels, *fn* is

the number of pixels that are falsely marked as background and *fp* is the number of pixels that are falsely detected as foreground.

Results & Findings

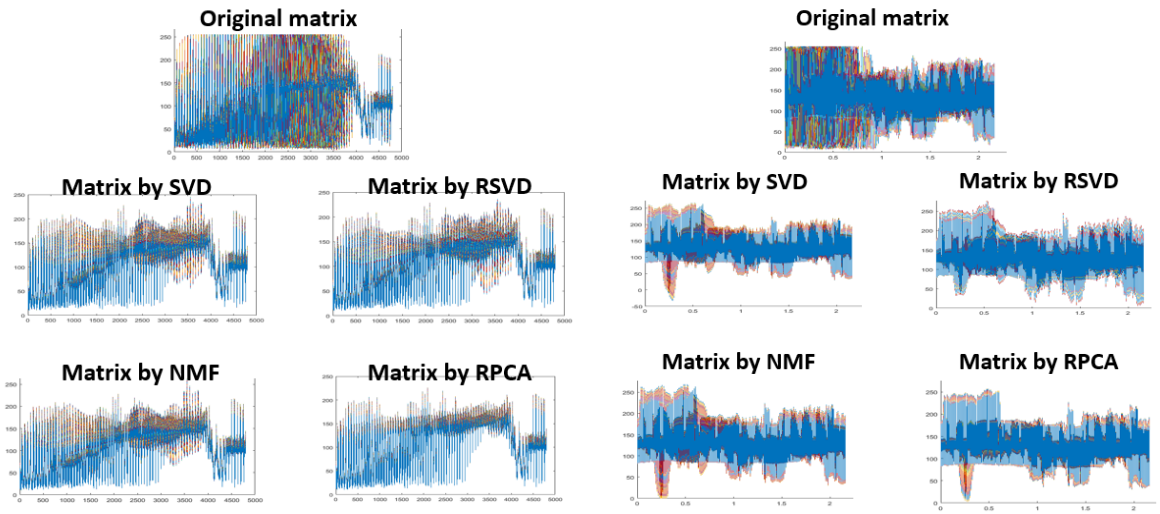
The algorithms were first implemented on the baseline dataset of a highway with cars passing (Case-A). The performance, as depicted in Figure 7, indicates a high precision and accuracy and an F-score of ~60%, with robust PCA having the highest F-Score of 64.16%. The marginally better performance of robust PCA is also illustrated in Fig. 6a, where it seems to extract the background of the video (in blue) with minimal foreground components. The algorithms were then applied to a video of people moving in a train station and people playing badminton that had camera jitters (Case-B). Similar performance measures were observed as shown in one of the frames of the output video in Fig. 5b. Here too, we noticed comparable performances across all four algorithms with high precision (~55%) but robust PCA reported particularly higher precision (85.6%) as compared to the other algorithms. This is also evident from matrix representation of Case-B in Fig. 6b.

Highway (Case-A) Train Station (Case-B) Canoe (Case-C)

Figure 5: Foreground detection of the a) baseline, b) camera jitter, and c) dynamic background videos from the CDnet2014 dataset. The first row of images is the original image, the second row is the image from the video with foreground removed and the last row depicts the video with only the foreground object I.e. cars, people and the canoe, respectively

In order to check for the performance of the algorithms on more challenging datasets, we applied the algorithms to a video with dynamic background (Case-C). The algorithms performed badly while detecting the movement of the water in the video. This can be observed in Fig. 5c where the foreground image includes the canoe and parts of the water body as well. Hence, while the precision was high, the recall of

the algorithms was low, resulting in an overall low f-score. Robust PCA reported the lowest f-score (21.1%) with other algorithms having nearly equivalent measures (~26%).



Highway (Case-A) **Badminton (Case-B)**

Figure 6: Illustration of the matrices depicting the extracted background of the videos through the four matrix decomposition techniques a) baseline video b) video with camera jitter

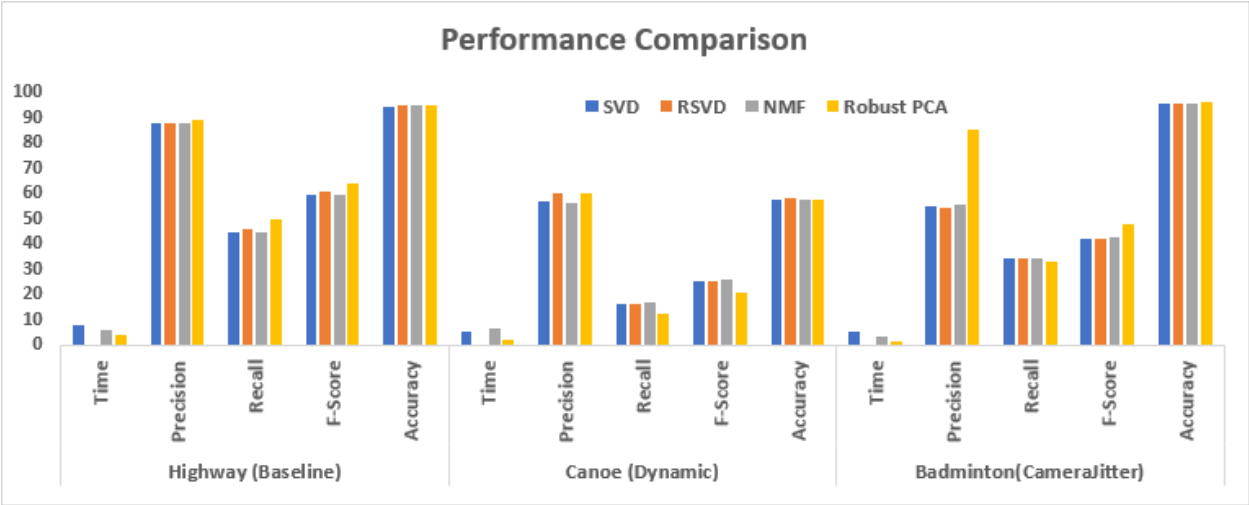


Figure 7: Comparison of the performance of the SVD, RSVD, NMF, and robust PCA across the baseline, camera jitter, and dynamic background video from the CDnet2014 dataset.

When comparing across time, rSVD recorded the lowest time to run on videos of ~500 frames, while SVD consumed the maximum time amongst the four algorithms. The comparison of performance of the four algorithms as measured by the five parameters (time, precision, recall, f-score and accuracy) are shown in Fig. 7.

Conclusion & Discussions

The comparison of the background subtraction algorithm offers quantitative insights on the performance of the current 'mainstream' approaches for background removal for video dataset. This study evaluated four popular algorithms for three different datasets (under both static and dynamic background settings); the results are compared by using 4 testing parameters along with computational time. No method performs best in all conditions; but the robust PCA yielded more superior results with marginal improvements measured with a slightly higher f-score under the static (Case-A) and jitter (Case-B) background conditions. The f-score was observed to be the most reliable parameter for performance comparison since it reflects the balance between precision and recall, whereas accuracy is less optimal for comparison as the real-life video frames typically present an imbalance dataset.

References

Babwin. D. Cameras make chicago most closely watched U.S. city. Huffington Post, April 6 2010

Bucak, S. S., Günsel, B., & Gursoy, O. (2007, June). Incremental Non-negative Matrix Factorization for Dynamic Background Modelling. In *PRIS* (pp. 107-116).

Grosek, J., & Kutz, J. N. (2014). Dynamic mode decomposition for real-time background/foreground separation in video. *arXiv preprint arXiv:1404.7592*.

Guo, L., Xu, D., & Qiang, Z. (2016). Background subtraction using local svd binary pattern. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops* (pp. 86-94)

Halko, N., Martinsson, P. G., & Tropp, J. A. (2011). Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, *53*(2), 217-288.

He, J., Balzano, L., & Szlam, A. (2012, June). Incremental gradient on the grassmannian for online foreground and background separation in subsampled video. In *2012 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1568-1575). IEEE.

Lee, D., Seung, H., 2001. Algorithms for non-negative matrix factorization. Adv. Neural Inform. Process. Systems 13, 556–562

Tian, Y. L., Lu, M., & Hampapur, A. (2005, June). Robust and efficient foreground analysis for real-time video surveillance. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern*

Y. Wang, P.-M. Jodoin, F. Porikli, J. Konrad, Y. Benezeth, and P. Ishwar, *CDnet 2014: An Expanded Change Detection Benchmark Dataset*, in Proc. IEEE Workshop on Change Detection (CDW-2014) at CVPR-2014, pp. 387-394. 2014