Ukrainian Catholic University

Faculty of Applied Sciences

Business Analytics

Background Subtraction

Linear Algebra Final Report

Authors:
Vitalii Sharaiev
Oleksandra Symko
Andrii Hlushko

Supervisor:
Nataliya Terlych

22 April 2024

Abstract

The problem of separating backgrounds and moving objects is an important part of modern photo and video processing. As digital technologies are rapidly evolving, this improves the quality of photo and video materials, and thus increases the amount of memory required to store and process the matrices that represent the corresponding images or videos. Thus, in this study, the authors try to solve the problem of background subtraction in a computationally efficient way. For background estimation, low-rank matrix approximations are used, which are obtained by the computationally efficient Randomized SVD algorithm. This report provides an explanation of the Randomized SVD algorithm and the results of background approximation. In conclusion, the authors argue that SVD is a powerful tool in general, but it is best used for photo and video preprocessing.

Keywords— Background Subtraction - Randomized Singular Value Decomposition

1 Introduction

With static cameras the background, like buildings, trees, roads, yards, rooms etc., stays mostly constant over a series of frames and may vary due to illumination or weather conditions, whereas the foreground consisting of objects of interest which might be cars or humans or some other objects depending on application, causes differences in image sequences. Background subtraction aims to distinguish between foreground and background based on previous image sequences and eliminates the background from newly incoming frames, leaving only the moving objects contained in the foreground.

1.1 Motivation

Background subtraction is a widely used approach in computer vision for segmenting objects of interest from the background within a scene. It is particularly useful in scenarios where the objects of interest are moving, and the background remains relatively static. Essentially, background subtraction involves comparing each new frame of a video to a model of the background and identifying differences, which are presumed to be caused by moving objects.

Applications of background subtraction are diverse, covering areas such as:

- Surveillance as automated monitoring of secure areas, detection of unauthorized entries, and triggering alarms or notifications.
- Traffic control as detection of vehicles traffic scenes, which can be used for monitoring of traffic flow, and identification of traffic violations.

- Object tracking as background subtraction helps to maintain the focus on moving objects across the frames of a video.
- Augmented Reality (AR) as background subtraction helps to superimpose virtual objects into real-world scenes accurately.

1.2 Background Subtraction as Optimization Problem

Under assumption that camera is static the background subtraction problem can be formulated as an optimization one. Suppose there is a sequence of frames stored in matrix $\mathbf{A} \in \mathcal{R}^{d\times n}$, where d represents the number of pixels and n represents the number of frames, then background-foreground separation can be modeled as decomposing $\mathbf{A}$ into low-rank matrix $\mathbf{A_k}$, which represents background and sparse matrix $\mathbf{M}$, which represent moving objects. This leafs to the following problem:

$$min||\mathbf{A} - \mathbf{A_K}||_F$$

which, under the best approximation of $\mathbf{A_k}$ should, give a movement matrix $\mathbf{M}$

2 Data Description

Before application of any algorithm, video should be correctly preprocessed. To obtain a matrix that represents a video, the team decided to extract each frame of the video according to the video's per second frame rate. Then each frame was converted into grayscale and flattened into a column vector $v_k \in \mathcal{R}^{d\times 1}$, where d stands for the number of pixels. Each flattened vector then forms a video matrix, columns of which represent a flattened frames in correct chronological order.

Example of obtained matrix representing video.

Algorithm 1 Function to construct a matrix from video frames

Result: Matrix constructed from frames
Function `construct_matrix`(fps, $duration$):
    frames ← empty list
    for $i \leftarrow 1$ **to** $fps \times duration$ **do**
        frame ← retrieve frame(i/fps)
        frame ← grayscale(frame)
        frame ← flatten(frame)
        append frames with frame
    end
    return matrix(frames transposed)

3 Solution Description

Since the team deals with video samples that have static background, then background may be considered as dominant part of the whole video sample. Thus to obtain the best solution for the optimization problem, it is necessary to construct such a low-rank approximation matrix that describes only dominant pattern of the video.

For such matrix approximation may be chosen a rank-1. Here comes the brief explanation[1]: Matrices $\mathbf{AA^T}$ and $\mathbf{A^TA}$ consist of inner products of $\mathbf{i}$-th and $\mathbf{j}$-th columns, which represent $\mathbf{i}$-th and $\mathbf{j}$-th frames of the video - $< \mathbf{a_i}, \mathbf{a_j} >$. The inner product may be considered as the measure of similarity between two vectors. Out of eigenvectors of these matrices then are constructed matrices $\mathbf{U}$ and $\mathbf{V^T}$ appropriately of SVD decomposition and vectors of these matrices are called left singular and right singular vectors. The diagonal entries of $\mathbf{\Sigma}$ are the square roots of overlaping eigenvalues of matrices $\mathbf{AA^T}$ and $\mathbf{A^TA}$ and are called singular values. Singular values are sorted in descending order as well as eigenvectors are sorted to corresponding singular values. Taking into account statements above, the product of the first singular value and its corresponding singular vectors $\sigma_\mathbf{1}\mathbf{u_1}\mathbf{v_1}^\mathbf{T}$ forms a matrix that consists of frames that posses only dominant pattern for each particular frame. Since background is expected to be constant throughout the whole video sample, then dominant pattern formed by rank-1 approximation will model background.

To efficiently obtain low-rank approximation Randomized SVD is preferred to use. Also to ensure that the rank-1 approximation models clear background without shadows, the methods of oversampling [2] and power iterations [2] will be applied.

After the low-rank approximation was obtained, each column of such approximated matrix represent background model at a particular point of time. To obtain the best background approximation, the team decided to take mean within all background models.

Remark 1 *Oversampling is a concept used to increase probability to capture all the features that matrix posses. Such parameter may be calculated from the inequality provided derived according to Johnson-Lindenstrauss Lemma, which gives an error bounds that* $\mathbf{p} > \frac{\mathbf{log(n)}}{\epsilon^\mathbf{2}}$*, where* $\mathbf{p}$ *is the number of columns in random projection matrix,* $\mathbf{n}$ *- number of observations (in case of video, number of pixels) and* ϵ *- desired error level*

Remark 2 *Power iterations is a tool to compute dominant eigenvalues and is used when eigenvalues decay slowly. The computation of power iterations:* $\mathbf{X^q} = (\mathbf{XX^T})^\mathbf{q}\mathbf{X} = \mathbf{U\Sigma^{2q-1}V^T}$

4 Randomized Singular Value Decomposition

4.1 Notion

The first step of rSVD is to construct an orthogonal matrix that will span a column space of $\mathbf{A}$: $\mathbf{Q} \in \mathcal{R}^{\mathbf{d\times k}}$, where $\mathbf{d}$ is the number of pixels and $\mathbf{k} = \mathbf{r} + \mathbf{p}$, where $\mathbf{r}$ is the rank and $\mathbf{p}$ is an oversampling parameter.

The next step, if such matrix $\mathbf{Q}$ exists, compose matrix $\mathbf{B} = \mathbf{Q^TA}$, $\mathbf{B} \in \mathcal{R}^{\mathbf{n\times k}}$, which is a low-rank approximation of initial matrix $\mathbf{A}$. $\mathbf{n}$ in terms of background subtraction stands for the number of frames that video sample is composed of.

Then another problem of optimization arises. It is necessary to compose such matrix $\mathbf{Q}$ that:

$$min||A - QQ^TA||_F$$

4.2 Random Projection Matrix

Random projection matrix

$$\mathbf{\Omega} = \begin{bmatrix} | & | & & | \\ \mathbf{w}_1 & \mathbf{w}_2 & \cdots & \mathbf{w}_k \\ | & | & & | \end{bmatrix}$$

where $\mathbf{w_i} \in \mathcal{R}^{\mathbf{n}\times\mathbf{1}}$ is a vector each occurrence of which is a realisation random variable with standard normal distribution . Is is known that in high dimensions, the probability that $< \mathbf{w_i}, \mathbf{w_j} > \approx \mathbf{0}$ is almost one.

Proof why random vectors tend to be almost orthogonal in high dimensional space.

4.3 Orthogonal basis

Then, it is appropriate to form a matrix $\mathbf{Y} = \mathbf{A\Omega}$, $\mathbf{Y} \in \mathcal{R}^{\mathbf{m}\times\mathbf{k}}$. After the $\mathbf{QR}$ on matrix $\mathbf{Y}$, the team obtains an orthogonal matrix $\mathbf{Q} \in \mathcal{R}^{m\times k}$ which spans column space of $\mathbf{Y}$. Since matrix $\mathbf{Y}$ is an approximation of matrix $\mathbf{A}$, then matrix $\mathbf{Q}$ also spans column space of matrix $\mathbf{A}$.

4.4 Components of SVD

Since matrix $\mathbf{Q}$ exists, it is now possible to construct matrix $\mathbf{B} = \mathbf{Q^T A}$. The next step is to perform SVD on obtained matrix $\mathbf{B}$

$\mathbf{B} = \mathbf{\tilde{U}\Sigma V^T}$

Since $\mathbf{B} = \mathbf{Q^T A}$, then $\mathbf{A} = \mathbf{QB} = \mathbf{Q\tilde{U}\Sigma V^T}$. Now to obtain singular value decomposition of initial matrix $\mathbf{A}$ it is needed to calculate matrix $\mathbf{U} = \mathbf{Q\tilde{U}}$. (Here: $\mathbf{U} \in \mathcal{R}^{\mathbf{d}\times\mathbf{k}}$, $\mathbf{\Sigma} \in \mathcal{R}^{\mathbf{k}\times\mathbf{k}}$, and $\mathbf{V^T} \in \mathcal{R}^{\mathbf{k}\times\mathbf{k}}$)

Thus, the low-rank approximation of initial matrix $\mathbf{A}$ has been computed without storing the whole matrix $\mathbf{A}$ or computationally inefficient or even impossible caluclations with full-rank matrix $\mathbf{A}$.

More theoretical datails on randomized singular value decomposition here.

Algorithm 2 Randomized SVD computation

```
Function randomized_svd(matrix M, r, q, p):
    ncolumns ← number of columns in M
    Ω ← random normal matrix of size ncolumns by (r + p)
    Y ← M × Ω
    for k ← 0 to q − 1 do
        Y ← M × (M^T × Y)
    end
    Q, R ← QR decomposition of Y, mode='reduced'
    Y ← Q^T × M
    Ũ, S, V ← SVD of Y, mode='reduced'
    U ← Q × Ũ
    return U, S, V
```

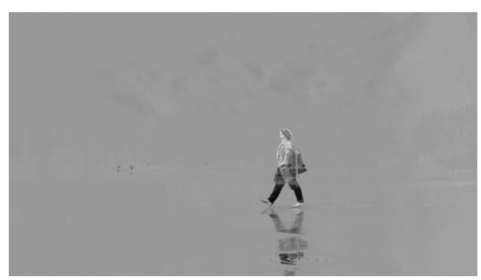
5 Results

Since the team clearly understands the type of data it needs, then the data set was composed on our own. The were several locations on the UCU Campus captured.

To review source code visit the link.

(a) Approximated background

(b) Frame from the video



(c) Subtracted background frame

Figure 1: Background Segmentation

To observe more examples look here.

6 Conclusions

As one can observe the SVD is for sure a very robust tool. As depicted on the results, it is possible to obtain a very good model of the background. But nowadays there exist much efficient approaches to calculate such background model. Moreover, approach that depends on SVD has significant drawbacks in terms of calculation optimization. Even introducing the randomized SVD which considerably speeds up the process for calculating the decomposition, it is still comparatively slow while dealing with megapixel images or frames, thus it is not a good idea to use SVD for real-time application. But all in all, SVD provides a great preprocessing facility of the images before applying some more advanced mathematical methods.

Project sourced on GitHub

7 References

1. Steven L. Brunton, J. Nathan Kutz, Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control, Cambridge University Press; 1st edition (April 18, 2019) p. 12-13

2. Steven L. Brunton, J. Nathan Kutz, Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control, Cambridge University Press; 1st edition (April 18, 2019) p. 44-45

3. Johnson-Lindenstrauss Lemma

4. Orthogonality expected at higher dimensions by Doug Turnbull

5. Why are randomly drawn vectors nearly perpendicular in high dimensions

6. Applied Linear Algebra: Randomized Linear Algebra by Nathan Kutz