MULTI-TEMPORAL FOREGROUND DETECTION IN VIDEOS

Mariano Tepper[†], *Alasdair Newson*[†], *Pablo Sprechmann*[‡], *Guillermo Sapiro*[†]

[†]Department of Electrical and Computer Engineering, Duke University
[‡]Courant Institute, New York University

ABSTRACT

A common task in video processing is the binary separation of a video's content into either background or moving foreground. However, many situations require a foreground analysis with a finer temporal granularity, in particular for objects or people which remain immobile for a certain period of time. We propose an efficient method which detects foreground at different timescales, by exploiting the desirable theoretical and practical properties of Robust Principal Component Analysis. Our algorithm can be used in a variety of scenarios such as detecting people who have fallen in a video, or analysing the fluidity of road traffic, while avoiding costly computations needed for nearest neighbours searches or optical flow analysis. Finally, our algorithm has the useful ability to perform motion analysis without explicitly requiring computationally expensive motion estimation.

Index Terms— Video, foreground detection, robust PCA,

1. INTRODUCTION

The detection and separation of videos into background and moving foreground is a critical in many computer vision applications, and therefore much research has been dedicated to this problem. Traditionally, the classification of video content as background or foreground is completely binary. However many detection problems require a finer characterisation of the foreground. For instance, we may wish to detect people in a video who are immobile for a certain period of time (if they have fallen, or are unconscious for example). In this case, a standard foreground detection would not distinguish between moving people, who do not need help, and still people, who require aid.

Accordingly, we introduce the concept of foreground detection at multiple timescales, and propose an online algorithm to achieve this. At its core, this algorithm uses *robust principal component analysis* (RPCA), introduced by Candès et al. [1]. The authors consider the problem of separating a matrix into two components: a low-rank component (the background), and a sparse component (the foreground). This is formulated as a convex optimisation problem.

To detect foreground objects at multiple timescales, we define several temporal windows in which we estimate background and foreground using an RPCA based approach. Then, each pixel in the video is assigned a discrete label, corresponding to its characteristic timescale. This is the minimum duration required for the pixel to be considered as foreground. Finally, in order to increase spatial robustness and coherence of the approach, we smooth this labelling using a technique based on graph cuts [2]. For an example of this labelling, see Figure 1. Others are presented further on.

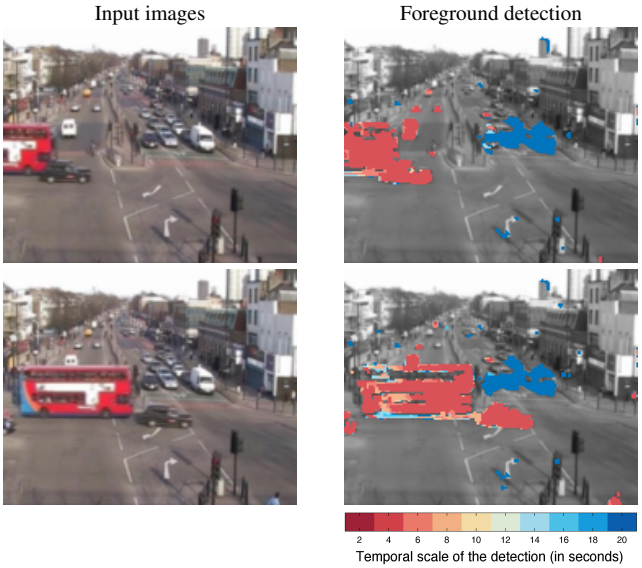

Fig. 1: **Automatic detection of still and moving traffic**. The multi-temporal foreground is highlighted in colour in the images on the right. Blue corresponds to longer-term foreground, and red corresponds to shorter-term foreground. Cars stopped at the traffic lights are highlighted in blue, whereas those which are in movement are coloured in red.

The resulting detection highlights foreground objects at user-defined timescales. This is useful for a variety of applications, such as detecting fallen people or automatically analysing automobile traffic. The detection is performed *without any explicit motion estimation*, which can be costly or challenging in certain scenarios, such as those dealing with poor quality surveillance cameras. Furthermore, the algorithm has a very small memory footprint, thanks to the use of

online RPCA. These characteristics of the approach make it well suited for online applications.

Related work. Much research has been done in the area of background estimation and background/foreground separation. Wren et al. associate a Gaussian random variable for each pixel [3]. A very commonly used approach is that of Stauffer and Grimson [4], who model each pixel of the background as a weighted sum of Gaussian random variables (a Gaussian mixture model), which lends greater robustness and flexibility to their method. Other authors have modelled the background using richer information than just pixel colour. Heikkila and Pietikainen [5] used local binary patterns to model the texture of the background. Yao and Odobez [6] employed both colour and texture to describe the background. A certain portion of the research on background estimation is concerned with shadows, which are often incorrectly identified as foreground [7, 8]. More recently, Candès et al. [1] considered the problem of low-rank and sparse matrix decomposition, with one major application being background estimation. A great advantage of this approach is that the foreground of a video is explicitly present in the minimisation, whereas approaches such as [4] require an additional thresholding operation, adding further parameters to be tuned.

2. MULTI-TEMPORAL FOREGROUND DETECTION

Let us introduce some notation. We consider a *data matrix* $\mathbf{X} \in \mathbb{R}^{m\times n}$ which represents the video data. Each column of $\mathbf{X}$ corresponds to a vectorised frame (containing m pixels) of the video with n frames. We wish to decompose $\mathbf{X}$ as: $\mathbf{X} = \mathbf{L}+\mathbf{O}+\mathbf{E}$, where $\mathbf{L}$ is a low-rank matrix (the background), $\mathbf{O}$ is a sparse matrix (the foreground) and $\mathbf{E}$ is an error matrix. RPCA performs this decomposition by solving the following convex optimisation problem

$$\min_{\mathbf{L},\mathbf{O}\in\mathbb{R}^{m\times n}} \tfrac{1}{2}\|\mathbf{X}-\mathbf{L}-\mathbf{O}\|_F^2 + \lambda_*\|\mathbf{L}\|_* + \lambda\|\mathbf{O}\|_1, \quad (1)$$

where $\|\cdot\|_F$ is the Frobenius matrix norm, $\|\mathbf{L}\|_* = \sum_i \sigma_i(\mathbf{L})$ is the *nuclear norm* of the matrix $\mathbf{L}$ and $\sigma_i(\mathbf{L})$ is the i^{th} singular value of $\mathbf{L}$, $\|\cdot\|_1$ is the ℓ^1 matrix norm, and λ_* and λ are scalar optimisation parameters.

2.1. Online RPCA

Given our target applications, we need to deal with frames as they appear in the video stream. Therefore, we want to avoid computing the solution to the complete optimisation problem when each frame arrives. Also, the previous video information can presumably be reused to speed up computation. We do this using the framework presented in [9], which we refer to here as *online* RPCA. This is designed to deal with situations where data arrives sequentially.

It can be shown [10, 11] that if an upper bound q on the rank of a matrix $\mathbf{L}$ exists, then the nuclear norm can be reformulated as the solution of the following optimisation problem

$$\min_{\substack{\mathbf{U}\in\mathbb{R}^{m\times q}\\ \mathbf{S}\in\mathbb{R}^{q\times n}}} \tfrac{1}{2}\|\mathbf{U}\|_F^2 + \tfrac{1}{2}\|\mathbf{S}\|_F^2 \qquad \text{subject to} \qquad \mathbf{L}=\mathbf{U}\mathbf{S}. \quad (2)$$

In the applications which we are considering, it is reasonable to set an upper bound q. By combining equations (1) and (2), it is possible [12] to formulate the RPCA minimisation problem as

$$\min_{\mathbf{U},\mathbf{S},\mathbf{O}} \tfrac{1}{2}||\mathbf{X}-\mathbf{U}\mathbf{S}-\mathbf{O}||_F^2 + \tfrac{\lambda_*}{2}||\mathbf{U}||_F^2 + \tfrac{\lambda_*}{2}||\mathbf{S}||_F^2 + \lambda||\mathbf{O}||_1. \quad (3)$$

When data arrives sequentially, as in a video stream, this optimisation problem can be solved online using an alternating minimisation scheme [13].

Let $\mathbf{x}^t \in \mathbb{R}^m$ denote the current data frame, and $\mathbf{U}^t \in \mathbb{R}^{m\times q}$, $\mathbf{s}^t \in \mathbb{R}^q$, and $\mathbf{o}^t \in \mathbb{R}^m$ the desired decomposition matrices. We first find $\mathbf{s}^t$ and $\mathbf{o}^t$, given the previous estimate $\mathbf{U}^{t-1}$, as the solution of

$$\min_{\substack{\mathbf{s}\in\mathbb{R}^q\\ \mathbf{o}\in\mathbb{R}^m}} \tfrac{1}{2}\|\mathbf{x}^t-\mathbf{U}^{t-1}\mathbf{s}-\mathbf{o}\|_F^2 + \tfrac{\lambda_*}{2}\|\mathbf{s}\|_2^2 + \lambda\|\mathbf{o}\|_1. \quad (4)$$

We solve this using Algorithm 1. Then, $\mathbf{U}^t$ is found as the solution of

$$\mathbf{U}^t = \operatorname*{argmin}_{\mathbf{U}} \sum_{j=1}^{t} \tfrac{1}{2}\|\mathbf{x}^j-\mathbf{U}\mathbf{s}^j-\mathbf{o}^j\|_2^2 + \tfrac{\lambda_*}{2}\|\mathbf{U}\|_2^2. \quad (5)$$

This problem is a Tikhonov-regularised least squares which has the closed-form solution

$$\mathbf{U}^t = \left(\sum_{j=1}^{t}\mathbf{s}_j\mathbf{s}_j^{\mathrm{T}} + \lambda_*\mathbf{I}\right)^{-1}\sum_{j=1}^{t}(\mathbf{x}^j-\mathbf{o}^j)\mathbf{s}_j^{\mathrm{T}}. \quad (6)$$

Algorithm 1: Alternating minimization scheme for solving Equation (4)

input : Data $\mathbf{x}$, dictionary $\mathbf{U}$, parameters λ_* and λ.
output : Coefficient vector $\mathbf{s}$ and outlier vector $\mathbf{o}$.

$\mathbf{W} \leftarrow \mathbf{U}\left(\mathbf{U}^{\mathrm{T}}\mathbf{U}-\lambda_*\mathbf{I}\right)^{-1}\mathbf{U}^{\mathrm{T}}$; $\quad \mathbf{y}\leftarrow\mathbf{0}$; $\quad \mathbf{b}\leftarrow(\mathbf{I}-\mathbf{W})\mathbf{x}$

repeat

$\quad \mathbf{o}\leftarrow \mathrm{shrink}_\lambda(\mathbf{b})$ $\quad$ // $(\mathrm{shrink}_\lambda(\mathbf{b}))_i = \begin{cases} 0 & \text{if } b_i<\lambda \\ b_i-\lambda & \text{otherwise}\end{cases}$

$\quad \mathbf{b}\leftarrow \mathbf{b}+\mathbf{W}(\mathbf{o}-\mathbf{y})$

$\quad \mathbf{y}\leftarrow\mathbf{o}$

until *convergence*

$\mathbf{s} = \left(\mathbf{U}^{\mathrm{T}}\mathbf{U}-\lambda_*\mathbf{I}\right)^{-1}(\mathbf{x}-\mathbf{o})$

2.2. Multi-temporal analysis

To analyse the video with respect to multiple timescales, we compute K RPCA models, each over a temporal window $[t-t_k, t]$, with $t_1 < t_2 < \cdots < t_k < \cdots < t_K$. At each time

Algorithm 2: Online multi-temporal foreground detection algorithm

input : Data $\mathbf{X}$ at time t, long-term model $\mathbf{U}^{\infty}$, number of temporal windows K, length of each temporal window t_k, parameters λ_*, λ.
output : Multi-temporal detection map (ℓ_t).
Let $\mathbf{x}^t$ be frame t of $\mathbf{X}$
`/* Detect foreground with different time windows */`
for $k \leftarrow 1$ **to** K **do**
  Find $\mathbf{s}_k^t, \mathbf{o}_k^t$ using Equation (4) with $\mathbf{x}^t$, $\mathbf{U}_k^{t-1}$
  Find $\mathbf{U}_k^t$ using Equation (5), with $\{\mathbf{x}^j, \mathbf{s}_k^j, \mathbf{o}_k^j\}_{j=t-t_k,\dots,t}$
Find $\mathbf{s}_\infty^t, \mathbf{o}_\infty^t$ using Equation (4) with $\mathbf{x}^t$, $\mathbf{U}^{\infty}$
Associate a label $\ell^t(i)$ to each pixel i in $\mathbf{x}^t$ using Equation (7)
Smooth ℓ^t by solving Equation (8)

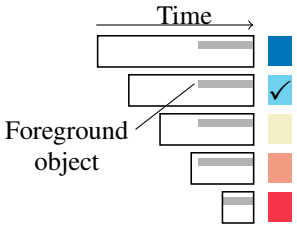

Fig. 2: **Visual illustration of the foreground detection algorithm.** A foreground object is associated with the shortest temporal window (indicated in light blue) at which it is considered to be part of the foreground. For any shorter windows, it will be part of the background.

step, the low rank and sparse representations of each window are updated with the current frame, using equations (4) and (6). Note that the summations in Equation (6) go from $t - t_k$ to t.

We also determine a long-term model $\mathbf{U}_\infty^t$, which represents the background with *no* foreground objects whatsoever. In our experiments, we establish this model using the whole video. However, in the applications which we are considering (fixed cameras with a constant field of view), it is reasonable to consider this background as known.

After decomposing the current frame $\mathbf{x}^t$, we establish a discrete labelling function $\ell : \mathbb{N}^2 \rightarrow \{1, \cdots, K\}$ which corresponds to the shortest temporal window at which a given pixel i is considered to be part of the sparse foreground. For longer timescales, this pixel will also be part of the foreground, while for shorter timescales it will belong to the background. For a visual illustration of this approach, see Figure 2. Formally,

$$\ell^t(i) = \min_{\mathbf{o}_k^t(i) \neq 0 \,\wedge\, \mathbf{o}_\infty^t(i) \neq 0} \{1, \dots, K\}. \tag{7}$$

Since the K models are completely independent, this process is entirely parallelisable, a key feature for an online algorithm.

Finally, we perform a spatial smoothing on these labels, to improve their robustness and spatial coherence. We do

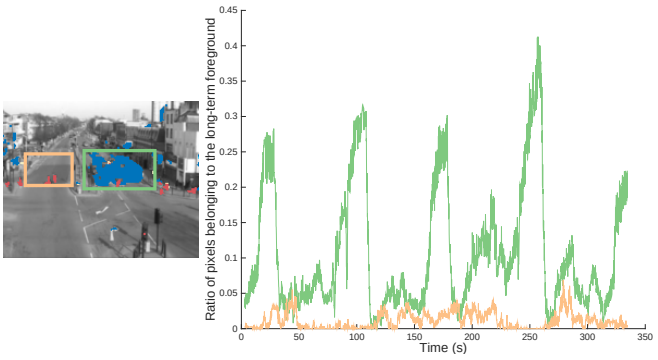

Fig. 3: **Automatic analysis of traffic.** Evolution of the ratio of long-term foreground pixels in two regions of the video. The peaks of the line in green, correspond to a red traffic light.

this using an efficient discrete optimisation technique based on graph cuts [2]. Accordingly, the final labelling is given by

$$\underset{\hat{\ell}}{\operatorname{argmin}} \sum_{i=1\dots m} \Big[E_d(\hat{\ell}(i)) + \sum_{j \in \mathcal{N}_i} E_s(\hat{\ell}(i), \hat{\ell}(j)) \Big], \tag{8}$$

where $\mathcal{N}_i$ is the 4-neighbourhood of the pixel location i and E_d and E_s are the following data and smoothness terms

$$E_d(\hat{\ell}(i)) = \begin{cases} \alpha & \text{if} \quad \hat{\ell}(i) \neq \ell(i) \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

$$E_s(\hat{\ell}(i), \hat{\ell}(j)) = \begin{cases} 1 & \text{if} \quad \hat{\ell}(i) \neq \hat{\ell}(j) \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

The scalar α is a constant penalty which we set to 1.5.

This final labelling represents the multi-temporal detection. The complete algorithm for the analysis of one incoming frame is presented in Algorithm 2.

Algorithm parameters. The main fixed parameters of the algorithm are the maximum rank of the low-rank background and the parameters of the optimisation problem. We use $q = 3$ as the maximum rank, which is reasonable for most videos. Similarly to the work of Zhou et al. [14], we set the optimisation parameters to $\lambda = \sigma\sqrt{2}$ and $\lambda_* = \sigma\sqrt{2n}$, where σ is an estimation of the standard deviation of the noise in the video.

The main tunable parameters are the temporal window sizes t_k, and the total number of temporal windows K. These parameters are highly dependent on the target application and video. Together, they determine the temporal granularity and the maximum timescale of the analysis. For a finer analysis, the window sizes should vary slowly. We indicate the timescales of our experiments in Section 3 and figures 1 and 4.

3. EXPERIMENTAL RESULTS

We now discuss some useful applications of the multi-temporal foreground detection. Also, since this work targets

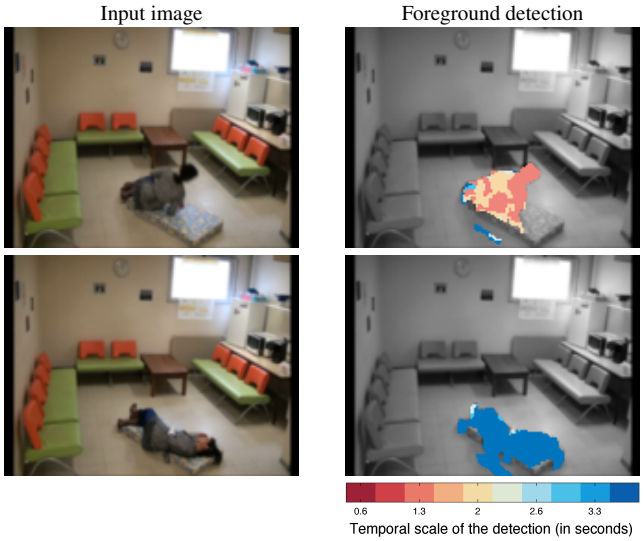

Fig. 4: **An example of detecting still persons**. In this application, we detect people who are still and may need assistance, as they have fallen.

online applications, we will discuss computation times. [1]

One application of this work is the automatic analysis of automobile traffic, in particular determining the fluidity of traffic. To test this application, we have used the "Qmul Junction" video from the work of Loy et al. [15], which shows the traffic of a busy junction. The corresponding foreground detection can be seen in Figure 1. We highlight the detected foreground in colour, whereas the background remains in greyscale. The labelling ℓ^t is represented with colours ranging from blue to red. Blue corresponds to longer-term foreground and red to short-term foreground. Thus, we are able to distinguish between moving objects, static objects, and objects which have recently come to a halt.

We perform an additional analysis on this example, as illustrated in Figure 3. We have selected two regions in the video centred near one of the traffic lights, and plotted the ratio of long-term foreground pixels in each area. It is immediately apparent that the ratio related to the window indicated in green evolves in a cyclic manner due to the presence of traffic lights, whereas the one coloured in orange presents no clear pattern. It would be relatively easy for a user to analyse this information and detect if any anomalies are occurring based on prior knowledge of the traffic light periodicity.

A very useful application of the proposed algorithm is the detection of immobile people, for example elderly persons who fall, or swimmers who are not moving. To illustrate this application, we have used the Fall Detection Dataset of Charfi et al. [16], which show videos of people falling and remaining still. Some results may be seen in Figure 4. We observe

[1] Our video results and Matlab code are available at `http://www.marianotepper.com.ar/multitemporal`.

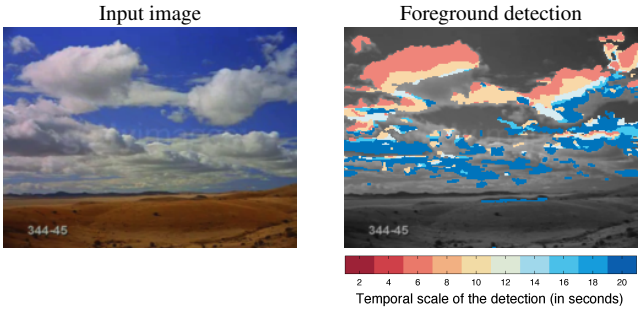

Fig. 5: **A time-lapse example**. Our algorithm is able to detect clouds moving at different speeds in the sky.

that the person is not detected while she is still moving, but is picked up in the long-term foreground when she remains still. By detecting objects in the correct timescale, a user could automatically detect people requiring help. An advantage of our approach is that we do not rely on human detection, which may not be robust to different poses and positions of humans.

Another interesting use for our algorithm is the analysis of time-lapse videos Figure 5. In such cases, we are interested in changes over specific periods of time. This could be useful in particular for analysing certain events in natural videos, or agricultural time-lapse footage.

An important goal of this work is to provide an online algorithm. Therefore, we wish for as low time and memory requirements as possible. On the "Qmul junction" example (see Figure 3), our algorithm detects the foreground for ten timescales in 0.8 seconds on average, with no parallelisation. We used a machine with an Intel i7 3.40 GHz processor and 32GB of RAM. Furthermore, our algorithm only requires to store one model $\mathbf{U} \in \mathbb{R}^{m \times q}$ for each timescale, which is much lower than storing a vector field for each frame as would be necessary with optical flow, for example.

4. CONCLUSION

In this paper, we have tackled the new problem of detecting foreground at multiple timescales in videos. The proposed algorithm produces a detection map which associates each pixel of the video with a characteristic timescale. This timescale represents the minimum duration required for the pixel to be detected as foreground. At the core of this approach lies the online RPCA algorithm. Finally, in order to improve spatial coherence, we smooth the labels using an efficient discrete optimisation technique. We have shown several important and useful applications of our multi-temporal analysis, such as automatic traffic analysis and the detection of fallen people. Our algorithm is fast and has a low memory footprint, making it very well adapted for online usage.

Acknowledgments: Work partially supported by NGA, DHS (APL), ONR, ARO, NSF, and AFOSR (NSSEFF).

5. REFERENCES

[1] E. Candès, X. Li, Y. Ma, and J. Wright, "Robust Principal Component Analysis?," *Journal of the ACM*, vol. 58, no. 3, pp. 1–37, June 2011.

[2] A. Delong, A. Osokin, H. Isack, and Y. Boykov, "Fast Approximate Energy Minimization with Label Costs," *International Journal of Computer Vision*, vol. 96, no. 1, pp. 1–27, July 2012.

[3] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland, "Pfinder: Real-Time Tracking of the Human Body," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 780–785, July 1997.

[4] C. Stauffer and W. Grimson, "Adaptive Background Mixture Models for Real-Time Tracking," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Aug. 1999, vol. 2, pp. 246–252.

[5] M. Heikkila and M. Pietikainen, "A Texture-Based Method for Modeling the Background and Detecting Moving Objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 4, pp. 657–662, Apr. 2006.

[6] J. Yao and J. Odobez, "Multi-Layer Background Subtraction Based on Color and Texture," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2007, pp. 1–8.

[7] J.-S. Hu, T.-M. Su, and S.-C. Jeng, "Robust Background Subtraction with Shadow and Highlight Removal for Indoor Surveillance," in *International Conference on Intelligent Robots and Systems*, Oct. 2006, pp. 4545–4550.

[8] J. Jacques, C. Jung, and S. Raupp Musse, "Background Subtraction and Shadow Detection in Grayscale Video Sequences," in *18th Brazilian Symposium on Computer Graphics and Image Processing*, Oct. 2005, pp. 189–196.

[9] P. Sprechmann, A. Bronstein, and G. Sapiro, "Learning Efficient Sparse and Low Rank Models," Tech. Rep., Dec. 2012, arXiv:1212.3631.

[10] Nathan Srebro and Adi Shraibman, "Rank, trace-norm and max-norm," in *Learning Theory*, 2005, pp. 545–560.

[11] B. Recht, M. Fazel, and P. Parrilo, "Guaranteed Minimum-Rank Solutions of Linear Matrix Equations via Nuclear Norm Minimization," *SIAM Review*, vol. 52, no. 3, pp. 471–501, Aug. 2010.

[12] G. Mateos and G. B. Giannakis, "Robust pca as bilinear decomposition with outlier-sparsity regularization," *IEEE Transactions on Signal Processing*, vol. 60, no. 10, pp. 5176–5190, 2012.

[13] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online Learning for Matrix Factorization and Sparse Coding," *The Journal of Machine Learning Research*, vol. 11, pp. 19–60, Mar. 2010.

[14] Z. Zhou, X. Li, J. Wright, E. Candès, and Y. Ma, "Stable Principal Component Pursuit," Tech. Rep., Jan. 2010, arXiv:1001.2363.

[15] C. Loy, T. Hospedales, T. Xiang, and S. Gong, "Stream-based joint exploration-exploitation active learning," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2012, pp. 1560–1567.

[16] I. Charfi, J. Miteran, J. Dubois, M. Atri, and R. Tourki, "Definition and Performance Evaluation of a Robust SVM Based Fall Detection Solution," in *Eighth International Conference on Signal Image Technology and Internet Based Systems*, Nov. 2012, pp. 218–224.