ANOMALY DETECTION IN MOVING-CAMERA VIDEOS WITH SPARSE AND LOW-RANK MATRIX DECOMPOSITIONS

Eric de Carvalho Jardim Silva

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia Elétrica.

Orientadores: Eduardo Antônio Barros da Silva
Sergio Lima Netto

Rio de Janeiro
Setembro de 2018

ANOMALY DETECTION IN MOVING-CAMERA VIDEOS WITH SPARSE AND LOW-RANK MATRIX DECOMPOSITIONS

Eric de Carvalho Jardim Silva

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Examinada por:

Prof. Eduardo Antônio Barros da Silva, Ph.D.

Prof. Sergio Lima Netto, Ph.D.

Prof. João Baptista de Oliveira e Souza Filho, D.Sc.

Prof. Marcos Craizer, D.Sc.

Prof. Hae Yong Kim, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
SETEMBRO DE 2018

Silva, Eric de Carvalho Jardim

Anomaly Detection in Moving-Camera Videos with Sparse and Low-Rank Matrix Decompositions/Eric de Carvalho Jardim Silva. – Rio de Janeiro: UFRJ/COPPE, 2018.

XII, 72 p.: il.; 29, 7cm.

Orientadores: Eduardo Antônio Barros da Silva
Sergio Lima Netto

Tese (doutorado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2018.

Referências Bibliográficas: p. 67 – 72.

1. sparse and low-rank matrix decomposition. 2. anomaly detection. 3. abandoned object detection. 4. moving camera. 5. video surveillance. 6. principal components analysis. 7. geometric domain-transformations. I. da Silva, Eduardo Antônio Barros *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

DETECÇÃO DE ANOMALIAS EM VÍDEOS DE CAMERAS MÓVEIS USANDO DECOMPOSIÇÕES EM MATRIZES ESPARSAS E DE BAIXO-POSTO

Eric de Carvalho Jardim Silva

Setembro/2018

Orientadores: Eduardo Antônio Barros da Silva
Sergio Lima Netto

Programa: Engenharia Elétrica

Apresentamos dois métodos baseados em decomposições esparsas que podem detectar anomalias em sequências de vídeo obtidas por câmeras em movimento. O primeiro método estima a união de subespaços (UdS) que melhor representa todos os quadros de um vídeo de referência (livre de anomalias) como uma projeção de baixo-posto mais um resíduo esparso. Em seguida, é realizada uma representação de baixo-posto do vídeo alvo (possivelmente anômalo) aproveitando a UdS e o resíduo esparso calculado a partir do vídeo de referência. As anomalias são extraídas após o pós-processamento destas informações residuais. Esse algoritmo fornece bons resultados de detecção, além de eliminar a necessidade de uma sincronização prévia dos vídeos. No entanto, essa técnica perde eficiência quando os vídeos de referência e alvo apresentam desalinhamentos mais graves entre si. Isso pode ocorrer devido a pequenos movimentos descontrolados da câmera e tremores durante a fase de aquisição. Para estender sua aplicabilidade, uma segunda contribuição é proposta a fim de lidar com esse possível desalinhamento. Isso é feito modelando a discrepância de pose de câmera entre os vídeos de referência e alvo com transformações geométricas agindo no domínio dos quadros do vídeo alvo. Um algoritmo completo de decomposição de matrizes é apresentado para realizar uma representação esparsa do vídeo alvo como uma combinação esparsa do vídeo de referência, levando em consideração as transformações que atuam sobre seus quadros. Nosso método é então verificado e comparado com técnicas de última geração com auxílio de vídeos de uma base desafiadora, apresentando os desalinhamentos em questão. Sob as métricas de avaliação usadas, o segundo método proposto exibe uma melhoria de pelo menos 16% em relação ao primeiro, e 22% sobre o método melhor avaliado logo em seguida.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

ANOMALY DETECTION IN MOVING-CAMERA VIDEOS WITH SPARSE AND LOW-RANK MATRIX DECOMPOSITIONS

Eric de Carvalho Jardim Silva

September/2018

Advisors: Eduardo Antônio Barros da Silva
Sergio Lima Netto

Department: Electrical Engineering

This work presents two methods based on sparse decompositions that can detect anomalies in video sequences obtained from moving cameras. The first method starts by computing the union of subspaces (UoS) that best represents all the frames from a reference (anomaly-free) video as a low-rank projection plus a sparse residue. Then it performs a low-rank representation of the target (possibly anomalous) video by taking advantage of both the UoS and the sparse residue computed from the reference video. The anomalies are extracted after post-processing this video with these residual data. Such algorithm provides good detection results while at the same time obviating the need for previous video synchronization. However, this technique looses its detection efficiency when target and reference videos presents more severe misalignments. This may happen due to small uncontrolled camera moviment and shaking during the acquisition phase, which is often common in real-world situations. To extend its applicability, a second contribution is proposed in order to cope with these possible pose misalignments. This is done by modeling the target-reference pose discrepancy as geometric transformations acting on the domain of frames of the target video. A complete matrix decomposition algorithm is presented in order to perform a sparse representation of the target video as a sparse combination of the reference video plus a sparse residue, while taking into account the transformation acting on it. Our method is then verified and compared against state-of-the-art techniques using a challenging video dataset, that comprises recordings presenting the described misalignments. Under the evaluation metrics used, the second proposed method exhibits an improvement of at least 16% over the first proposed one, and 22% over the next best rated method.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Anomaly detection in images and video sequences is a classical research problem in computer vision and related areas, which has direct applications in many tasks, ranging from domestic security and medical diagnosis to issues faced by industrial and military activities [1, 7]. The increasing number of applications and the necessity of precise results are raising the demand for alternative, less human-dependant solutions. Apart from being a regularly known problem, automatic anomaly detection still remains a difficult and challenging topic due to several complex issues such as camera pose, illumination, shadows, occlusions, weather conditions, camera jitter, and so on [12]. Some of these can drasticaly increase detection difficulty, possibly breaking with critical premises assumed by well stabilished techniques.

In several surveillance tasks, additional cameras should be employed to deal with the problem of multiple occlusions. Some activities, especially in cluttered environments like industrial plants and offshore oil platforms, usually require multiple viewpoints for proper inspection [2, 3]. Such a need is even greater in hazardous environments and when there are places that are difficult to access [31, 34, 35]. Increasing the number of cameras also increases the amount of video to be analyzed what can become unpractical in a large facility.

An interesting approach to deal with this issue is to monitor several viewpoints from a single moving camera. In practice, a conventional camera can be mounted on top of a moving platform (e.g. a car, robot, or drone), that takes the camera to the desired positions along a predefined trajectory. While this approach can significantly reduce the necessary number of cameras, it also enables the selection of specific points-of-view to be monitored at the same time that it allows the automation of repetitive inspections. These factors, allied to the widespread use of portable cameras, are spurring the interest in problems of surveillance and background/foreground separation using moving cameras [4–6, 8, 31]. Figure 1 shows an example of detection of moving objects by a mounted moving camera as described in [6].

Figure 1.1: An example of detection technique of moving objects with a moving camera. This technique tries to detect moving objects from different points-of-view. Images extracted from [6].

This study proposes new methods to robustly detect changes in moving cameras sequences by using special matricial factorizations. There are two types of video sequences that are fundamental to all algorithms. Namely, the *reference video*, that is previously validated as containing no anomalies, and the *target video*, a possibly anomalous sequence that will be compared to reference data. Figure 1.2 illustrates sample still frames taken from these two kinds of sequences.

(a) (b)

Figure 1.2: (a) a supposedly normal image, used as reference; (b) similar image with an abandoned object (pink bottle).

The first proposed method uses principal subspace analysis and sparse representation to extract any anomalous information from video sequences. The process is divided into two stages, respectively using the reference and target videos sequences packed as matrices. First, in the *modeling stage*, the reference matrix is decomposed into the sum of a low-rank representation component plus a sparse residual. These two terms are used as input the *detection stage*, together with the target matrix (i.e., the matrix that is being tested for changes) in order to isolate the changes in a resulting matrix. During this phase, two additional sparse plus low-rank decompositions are performed. To enhance the results, a post-processing step can appended after the detection stage. A major advantage of this method is that no video synchronization or frame geometric registration is needed. The only basic assumption

for this algorithm to work efficiently is that the camera's poses and trajectories during the target and reference video acquisitions are similar, in such a way that the information in the frames of the target video is mostly contained in the frames from the reference video. Under these circumstances, the decomposition of the target frames as an sparse combination of the data processed from the reference frames can be achieved by linear convex optimization, as it will be described further.

Unfortunately, in real world scenarios, the camera trembles as it moves, and thus its pose and trajectory during recording of the reference video may present variations relative to its pose and trajectory during recording of the target video. A way to cope with these issues is to add to the optimization process an additional non-linear domain-transformation term, that will require an iterative linearization approximation. This domain transformation enables the method to find a better correspondence between reference and target video frames, thus yielding less false detections as result of the algorithm. Therefore, a second method is proposed with these modifications. The latter approach is also not based on a modeling stage, as in the first method, since in the second approach the target data can be compared directly with the reference data.

In this document we present two methods based on sparse decomposition framework in order to detect anomalies in video sequences obtained from moving cameras. This work has the following contributions:

- Two full anomaly detection algorithms that can be applied to videos acquired by a moving camera, without the need of previous video synchronization, based on sparse representation and a post-processing stage;
- A parameter tuning methodology using recursive subdivision search and by employing a K-fold cross-validation assessment;
- A qualitative and quantitative evaluation of the proposed algorithms, comparing their performance with competing state-of-the-art methods, using data from a very challenging dataset.

1.1 Thesis Organization

This document is organized as follows: Chapter 2 reviews the state-of-the-art techniques for linear low-rank modeling of high-dimensional data by using matrix decomposition with sparse representations and some variations that support geometric domain-transformations. The setup of the moving-camera surveillance problem is addressed in Chapter 3, reviewing the literature and some testing datasets. Chapter 4 introduces the problem of anomaly detection in videos acquired from moving

cameras and details the adaptation of one of these sparse representation-based techniques for the problem at hand. Chapter 5 then discusses how to make such a scheme robust to severe geometric misalignment between reference and target videos. The idea is to incorporate a transformation into the optimization problem associated to the sparse representation and validates the proposed methodology by presenting experimental results on a comprehensive video database, comparing it to the state-of-the-art. Finally, conclusions are drawn in Chapter 6 emphasizing our main contributions.

Chapter 2

Subspace Analysis and Matrix Decomposition

This chapter performs a modest literature review of the related techniques that inspired this research. Section 2.1 presents the evolution of linear modeling in high-dimensional data, from classical PCA to its modern robust derivatives that make use of optimization techniques. At last, Section 2.2 reviews the use of geometric domain-transformations as a mean to correct samples that are not perfectly aligned in PCA derived methods.

2.1 Sparse and Low-Rank Decomposition

When dealing with high-dimensional observations, a very popular and successful approach is to fit the data with a simplified, lower-dimensional model. More precisely, data is often assumed to lie approximately on some low-rank subspace, reducing model complexity and consequently increasing model robustness and simplifying further analysis and storage space. Unfortunately, real-world data usually comes from sensors, which may suffer from noise and other types of perturbations. Thus, these sensor readings can be modeled as the superposition of a low-rank component plus some kind of undesired corruption term. Mathematically, if $X \in \mathbb{R}^{m \times n}$ represents a matrix whose columns are these observed values, it can be modeled as

$$X = L + E, \tag{2.1}$$

where the columns of L represents a low-rank model and E is a matrix of perturbations. In practice, the problem of finding anomalies is reduced to the decomposition of X, isolating the spurious data into the perturbation component E.

In statistics, *principal component analysis* (PCA) is perhaps one of the most commonly known tool for analysis, and it is widely used in this kind of scenario. If

the rank of L is previously known and the entries of E are small, following independent and identical Gaussian distributions, the problem can be efficiently solved with PCA by simply performing a *singular value decomposition* (SVD) of X and projecting its columns into the subspace spanned by the r major left-singular vectors, sorted by its corresponding singular values in descending order.

Under these circumstances, the estimated subspace is optimal in the sense that it minimizes the mean squared reconstruction error of the columns of X, which allows us to rewrite the PCA as the optimization problem

$$\min_{L,E} ||E||_F \quad \text{subject to (s.t.)} \begin{cases} X = L + E \\ \text{rank}(L) \leq r \end{cases}, \tag{2.2}$$

where $||.||_F$ is the Frobenius norm of a given matrix.

A downside of this method is that, in many practical situations, r might not be known *a priori*. Even worse, the presence of large corruptions in E can significantly compromise the estimation of L. It is possible to demonstrate that a single corrupted entry can induce PCA to estimate a solution that is arbitrarily far from the correct one [22]. Hence, in order to generate decompositions with a broader range of usability, more error-tolerant approaches must be considered.

Robust principal component analysis

A very common practice in linear modeling is to use images with fixed pixel dimensions as long vectors. Techniques like *eigenfaces* are quite known in literature, which confirms the maturity of PCA in computer vision applications like face detection and recognition [19], for instance. In computer vision applications, it is desirable that the presence of some visual anomalies, like partially occluding objects, does not compromise the model accuracy. The robust PCA (RPCA) [22] presents an interesting solution to this issue by forcing a sparse error term E, that is, with most of its entries equal to zero. In the spirit of Eq. (2.1), it is fair to assume that E is sparse, since an occluding object will only affect part of the image pixels. On the other hand, the rank of L could be an intrinsic property of the underlying model and leaving it loose could be interesting for many purposes. To this end, Eq. (2.2) is modified so as to as generate the following minimization problem:

$$\min_{L,E} \text{rank}(L) + \lambda ||E||_0 \quad \text{s.t.} \quad X = L + E, \tag{2.3}$$

where $||.||_0$ is the l_0-norm (number of non-zero entries), and the parameter λ balances the sparsity of E and the rank of L. Notice that, unlike the case of Eq. 2.2, the rank of L is not constrained, and should be considered an intrinsic property of the data.

Unfortunately, this problem is intractable due to its combinatorial nature, and its convex relaxation is considered instead [25]:

$$\min_{L,E} \; ||L||_* + \lambda||E||_1 \quad \text{s.t.} \quad X = L + E, \tag{2.4}$$

where $||.||_*$ is the *nuclear norm* of a matrix, defined by $||A||_* = \text{trace}(\sqrt{A^T A})$. The main advantage of this latter formulation is that is tractable and it can be solved with high probability p under very weak conditions if λ is set to $1/\sqrt{\max(m,n)}$ [22], where m and n are the matrix dimensions.

The RPCA algorithm works very efficiently on scenarios with relatively static background, usually acquired by a static camera [18]. Two classical applications of robust PCA are in video surveillance and face recognition. In the first case, frames from a video sequence aquired from a relatively static surveillance camera are stacked into a matrix as flat vectors. The goal is to model the static background as the low-rank component L, including illumination changes and slow background variations. Foreground entities like fast moving objects generally occupy a small fraction of the image and should be isolated into the sparse component E, not contributing to the background model. In (a) and (b) from Figure 2.1, each row exhibits a sample of the original frame in the first column, the estimated low-rank component in the second column and the sparse error component in the third column. In matricial terms, each image in a given row corresponds to a given column (flattened image) of matrices X, L and E respectively. In (a) from Figure 2.1, notice that a standing man is modeled as background in L, since it appears still in all frames. Additionally, passers-by are captured as foreground objects in the E component. More accurately, what is captured in E is the pixel difference from the original frames to the estimated low-rank component for each frame, as can be seen in the third column of (a) and (b) in Figure 2.1. Looking at (b) in Figure 2.1 it is possible to see that RPCA can deal with illumination changes. In both examples, the technique seems to capture accurately the foreground objects as sparse-component errors.

In the face-recognition case, RPCA can be used to enhance a training set of face images by removing shadows, specularities or even accessories present in the training samples. It is known that convex, Lambertian objects under distant illumination lie near a nine-dimensional subspace [20]. Since human faces are not perfectly convex nor Lambertian, self-shadows and bright spots are quite common in face pictures, breaking the premises for the low-rank model. Fortunately, RPCA isolates shadows and specularities as sparse errors, as can be seen in the third column of (a) and (b) in Figure 2.2. The images of Figure 2.2 are organized in the same way as in Figure 2.1.

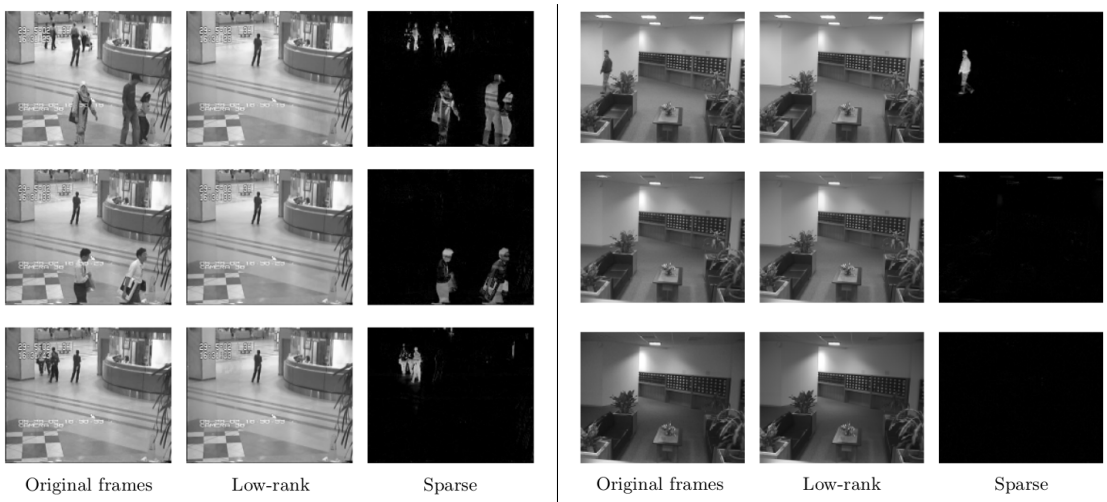


Figure 2.1: Sample frames of two experiments of RPCA used in video surveillance. In both (a) and (b), the first column represents the original frame (X), the second shows the estimated low-rank component (L) and the third contains the sparse component (E). Figure extracted from [22].

2.1.1 Robust Subspace Recovery (RoSuRe)

The use of convex optimization techniques was an undoubtedly important breakthrough in image and video analysis. Nevertheless, the pursuit for more general models has shown that the *union of subspaces* can be a more accurate representation of high-dimensional data when compared to the single subspace approach [42, 43]. So, the new question is: how to retrieve a set of unknown subspaces from possibly corrupted samples? This problem is clearly more complex than the single subspace modeling, once it is difficult to say, with no previous information, if a given sample is an outlier or it is representing a subspace. That is what the method known as *robust subspace recovery* (RoSuRe) intends to perform, taking into account all samples simultaneously.

Consider $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_k$ to be subspaces in $\mathbb{R}^m$ and let $L_1, L_2, \ldots, L_k$ be matrices where, for each L_j, the columns are uniformly sampled vectors from $\mathcal{S}_j$, assuming a sufficient sample density such that each column of L_j can be represented by the other columns with high probability. Since each L_j is *self-representative* by assumption, there exists a square coefficient matrix W_j with null diagonal which allows one to write $L_j = L_j W_j$. Let $\bigcup \mathcal{S}$ be the union of the $\mathcal{S}_j$ subspaces, that is

$$\bigcup \mathcal{S} = \bigcup_{j=1}^{k} \mathcal{S}_j.$$

Therefore, it is fair to say that $L = [L_1 | \ldots | L_k]$ is a self-representative sample matrix of $\bigcup \mathcal{S}$ in the same way each L_j is for $\mathcal{S}_j$. Indeed, one can easily build a

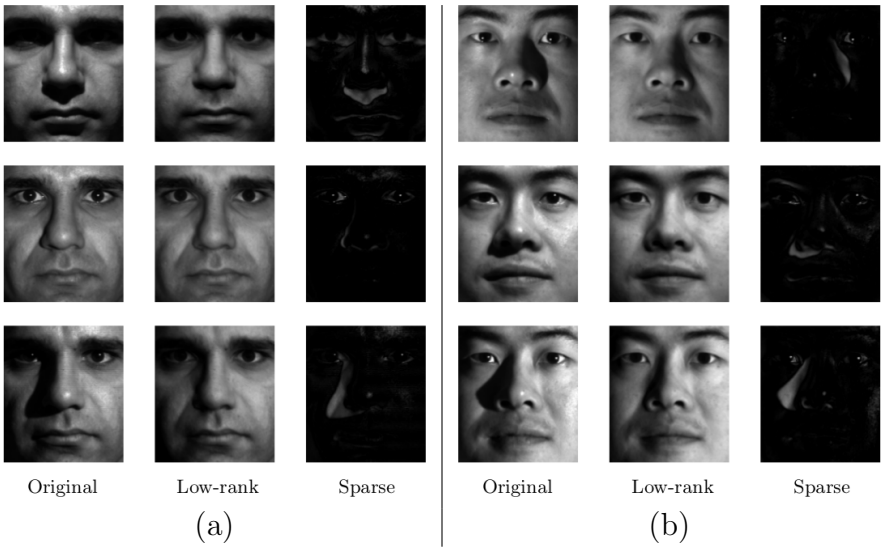


Figure 2.2: Sample frames of two experiments of RPCA used in face recognition. Shadows and specularities are extracted as sparse errors. Figure extracted from [22].

block-diagonal matrix $W = \text{diag}(W_1, \ldots, W_k)$, where W is clearly null diagonal, which provides

$$L = LW. \tag{2.5}$$

By construction, W is expected to be sparse in general and it reveals the undelying subspace structure represented by L, which is said to be *blockwise low-rank* in a basis induced by W. So, apart from possible corruptions, recovering $\bigcup \mathcal{S}$ from sampled data is equivalent to recovering L along with W. Thus, if X stacks n observations of $\bigcup \mathcal{S}$, the subspace recovery can be performed by decomposing

$$X = LW + E, \tag{2.6}$$

where E is again the matrix of perturbations. Notice that the (2.1) decomposition still holds by replacing (2.5) in (2.6), but since the structure present in W is important, the latter form should be considered.

The algorithm proposed in [32] assumes the sparsity of both W and E matrices to obtain the representation in (2.6). In the same fashion as in RPCA, it does so by solving the relaxation of a harder combinatorial optimization problem. The resulting relaxed problem

$$\min_{W,E} ||W||_1 + \lambda ||E||_1 \quad \text{s.t.} \begin{cases} X = L + E \\ LW = L \\ W_{ii} = 0 \end{cases}, \tag{2.7}$$

is not convex due to the bilinearity of W and E, but the global optimizer can be approximated by the *augmented Lagrangian multiplier* (ALM) method [32]. By

replacing L by $(X - E)$, the augmented Lagrangian function can be written as

$$\mathcal{L}(W, E, Y, \mu) = ||W||_1 + \lambda||E||_1 + \underbrace{\langle LW - L, Y\rangle + \frac{\mu}{2}||LW - L||^2}_{f(W,E)}, \tag{2.8}$$

where $f(W, E)$ is the differential part of $\mathcal{L}$, which is bilinear in W and E. Using the soft-threshold operator τ_α and letting $\hat{W}_k = I - W_k$, one can define the update steps of W and E:

$$\begin{aligned} W_{k+1} &= \arg\min_W ||W||_1 + f(W, E) \\ &= \tau_{\frac{1}{\mu\eta_1}}\left[W_k - \frac{1}{\eta_1}\nabla_W f(W_k, E_k)\right] \\ &= \tau_{\frac{1}{\mu\eta_1}}\left[W_k + \frac{1}{\eta_1}L_{k+1}^T\left(L_{k+1}\hat{W}_k - \frac{Y_k}{\mu_k}\right)\right], \end{aligned} \tag{2.9}$$

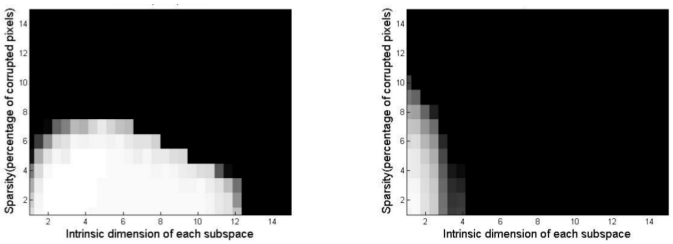
$$\begin{aligned} E_{k+1} &= \arg\min_E \lambda||E||_1 + f(W, E) \\ &= \tau_{\frac{\lambda}{\mu\eta_2}}\left[E_k - \frac{1}{\eta_2}\nabla_E f(W_{k+1}, E_k)\right] \\ &= \tau_{\frac{\lambda}{\mu\eta_2}}\left[E_k + \frac{1}{\eta_2}\left(L_{k+1}\hat{W}_{k+1} - \frac{Y_k}{\mu}\right)\hat{W}_{k+1}^T\right], \end{aligned} \tag{2.10}$$

where $\eta_1 \geq ||L||_2^2$ and $\eta_2 \geq ||\hat{W}||_2^2$, as summarized in Algorithm 1, which also includes the updates of Y and μ.

Algorithm 1 - Robust Subspace Recovery (RoSuRe) [32]

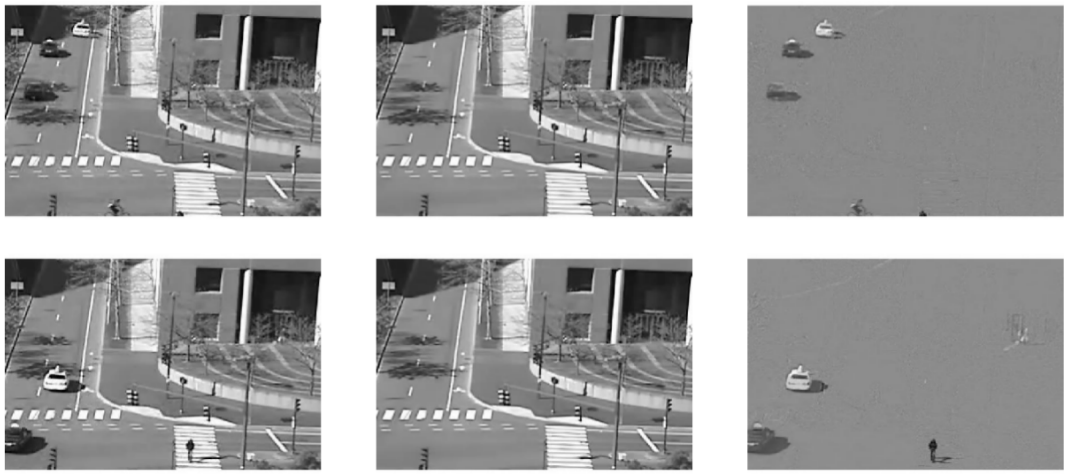
Input: $X, \lambda, \rho > 1, \eta_1, \eta_2$
while not converged **do**
  $L_{k+1} = X - E_k$
  $W_{k+1} = \tau_{\frac{1}{\mu\eta_1}}\left[W_k + \frac{1}{\eta^1}L_{k+1}^T\left(L_{k+1}\hat{W}_k - \frac{Y_k}{\mu_k}\right)\right]$
  $(W_{k+1})_{ii} = 0$
  $\hat{W}_{k+1} = I - W_{k+1}$
  $E_{k+1} = \tau_{\frac{\lambda}{\mu\eta_2}}\left[E_k + \frac{1}{\eta_2}\left(L_{k+1}\hat{W}_{k+1} - \frac{Y_k}{\mu_k}\right)\hat{W}_{k+1}^T\right]$
  $Y_{k+1} = Y_k + \mu_k\left(L_{k+1}W_{k+1} - L_{k+1}\right)$
  $\mu_{k+1} = \rho\mu_k$
end while

The RoSuRe method has shown to work successfuly with synthetic data, created by sampling vectors drawn from random union of subspaces with varying dimension added with random sparse error with varying percentage of corruption. Compared to RPCA, RoSuRe presents a broader range of correct recoveries when varying the intrinsic dimensions of the subspaces. The results can be seen in Figure 2.3 and more details can be found in the original paper [32], along with a discussion on sufficient conditions for exact recovery. Apart from successful tests with synthetic data,



RoSuRe Robust PCA

Figure 2.3: Overall recovery of RoSuRe and RPCA of samples drawn from random union of subspaces with varying dimension added with random sparse error with varying percentage of corruption. Success rate is plotted with grayscale color-code varying from black to white, respectively meaning 0 to 100 percentage of success. Results extracted from [32].



Original frame Background Foreground

Figure 2.4: Example of background subtraction with RoSuRe on traffic videos [32].

the RoSuRe method demonstrated strong potential in computer vision problems. Experiments with video surveillance and face clustering have shown its performance pairing up with state-of-art techniques. An example of background subtraction can be seen in Figure 2.4, in the same fashion of the RPCA experiment with video surveillance. A further experiment with a *virtual panning camera* opens the possibility of dealing with moving cameras. The samples of this test were actually generated by cropping a *sliding window* inside the original static surveillance video (see Figure 2.5). The window performs a periodic movement along the video frames as an attempt to mimic the actual movement of a panning camera, but ignoring visual parallax. Results are shown in Figure 2.6 and are slightly worse when compared with the static case. For this experiment, the coefficient matrix W computed with algorithm 1 exhibits the periodic frame correlation, as can be seen in the left image of Figure 2.7. Rearranging samples by the virtual camera position reveals an approximated blockwise diagonal structure as expected, as it is shown in the right

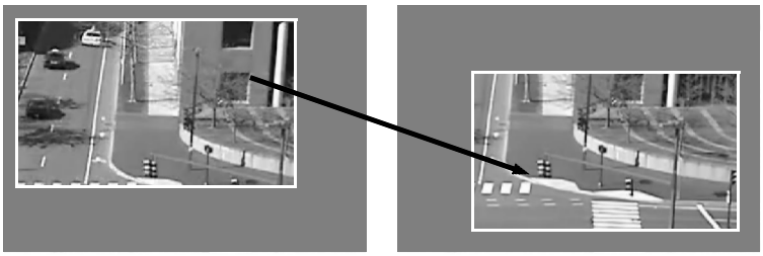


Figure 2.5: Virtual panning camera by cropping a sliding window inside original frame.

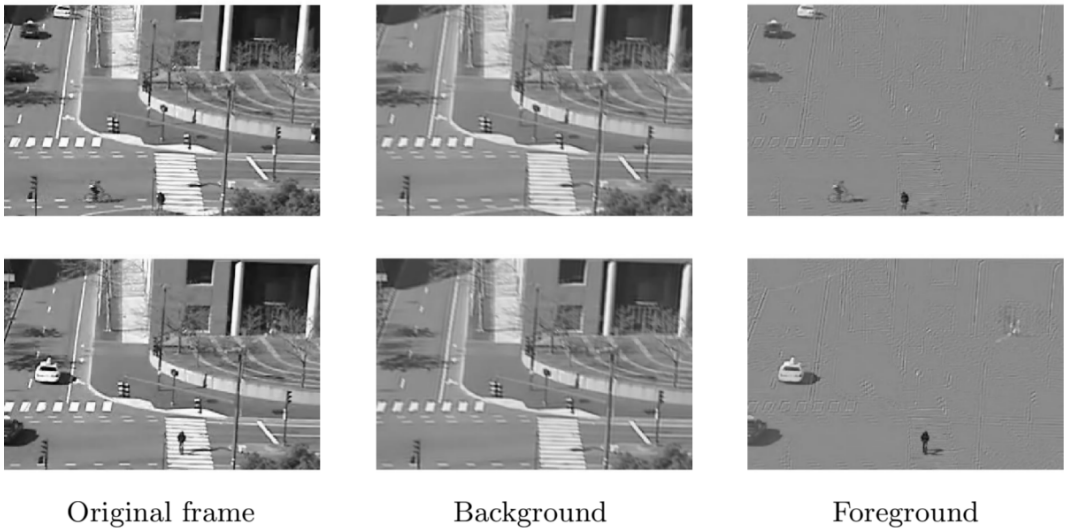


Figure 2.6: Example of background subtraction with RoSuRe with a virtual panning camera [32].

image of Figure 2.7. The success of this experiment is the main motivation for this work, since coping with moving cameras is the main interest of our investigation. The union of subspaces model seems to be a natural extension of the low-rank subspace model, since frames from different camera positions can be treated as samples drawn from different subspaces.

2.2 Domain Transformation

The use of linear modeling techniques applied to image samples was an important breakthrough in image analysis. However, the success of these methods is strongly attached to the pixelwise correlation of the sampling images. It is known that even small misalignments can break the linear structure that is being modeled, compromising the low-rank assumption upon the data. Unless sample images are previously registered or acquired under controlled conditions, misalignments are very common in practice, specially when dealing with moving cameras. To work around this issue, images can be considered to lie in a different geometric domain

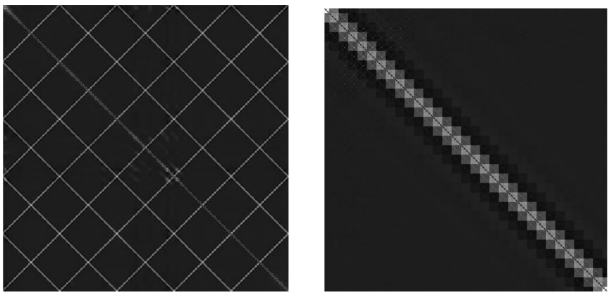


Figure 2.7: Absolute values of coefficient matrix W (left) and its rearrangement according to camera position (right) [32].

and the misalignments can be modeled as domain transformations. In this context, many techniques try to model simultaneously the data while they search for the best domain transformation that optimizes the parsimony of its representation [28, 44–47]. For future work purposes, two of these methods will be briefly reviewed.

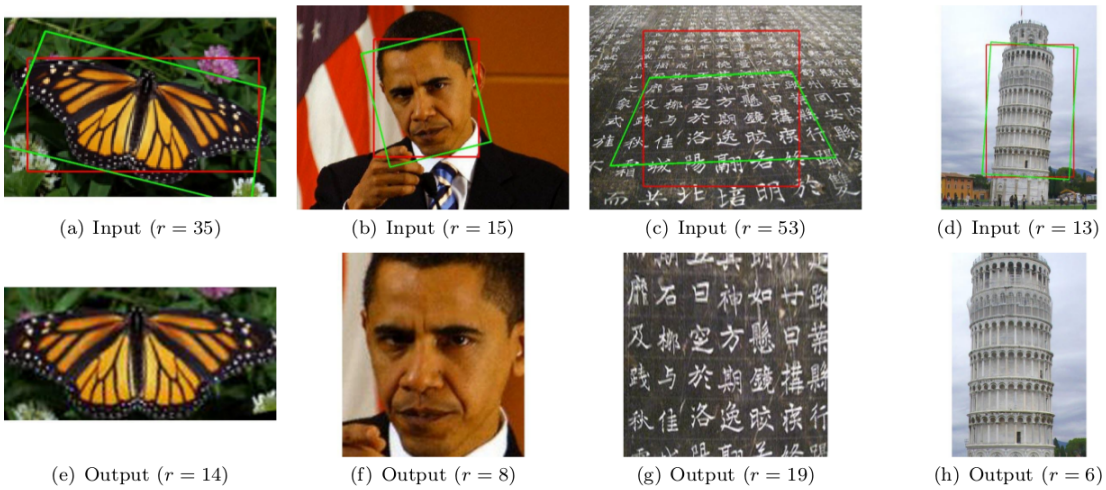
2.2.1 TILT: Transform Invariant Low-Rank Textures

In real world, most man-made objects and structures have a great amount of symmetry and redundancy. For example, the facade of a regular building could be approximately generated by a repeatable pattern of walls and windows. If it is under the correct alignment, the image of this same building could be generated by the repetition and combination of a few columns, possibly with very few corruptions, working as a low-rank texture. The idea behind TILT is to exploit this concept everywhere, searching for an alignment that reveals the low-rank texture of any desired image, with as few corruptions as possible.

However, it is obvious that different image alignments will lead to different textures, probably with different ranks. This raises the question: is there a "natural alignment" for an image, so it can be generated by a low-rank texture? To avoid the subjectivity of this question, TILT pursues the alignment that minimizes the rank of the generating texture, along with any possible corruption. More precisely, let I be an observed image, which under general assumption, is misaligned and corrupted. Let I^0 be its aligned low-rank texture and τ the domain transformation that corrects I. This means that $I \circ \tau$ is aligned with I^0, where the $\circ$ operator represents the action of the domain transformation τ on the image I. Thus, one can write $I \circ \tau = I^0 + E$, where E represents the corruptions. Finally, TILT can be defined as the folowing optimization problem:

$$\min_{I^0,E,\tau} \operatorname{rank}(I^0) + \gamma||E||_0 \quad \text{subject to} \quad I \circ \tau = I^0 + E. \tag{2.11}$$

In the same fashion as RPCA, this problem can be solved by convex relaxation. Gen-



(a) Input ($r = 35$) (b) Input ($r = 15$) (c) Input ($r = 53$) (d) Input ($r = 13$)

(e) Output ($r = 14$) (f) Output ($r = 8$) (g) Output ($r = 19$) (h) Output ($r = 6$)

Figure 2.8: Examples of TILT use in particular regions of the image. The red rectangle window on the top row are user-selected regions to be *tilted* and the green window are its transformed by τ. Notice the rank of the textures before and after TILT. Figure extracted from [44].

eraly, τ is chosen to be in a differentiable group of transformations like translations, rotations and homographies. Even τ being non-linear, it is possible to approximate the constraint by linearization around the current estimate of τ under small changes. These modifications are implemented in a extended version of the ALM algorithm [44, 45]. Examples of TILT use in particular image regions can be seen in Figure 2.8.

2.2.2 RASL: Robust Alignment by Sparse and Low-Rank Decomposition

As we said in section 2.2.1, sample misalignments can break the linearity assumption, compromising the low-rank property of the underlying model. The RASL technique tries to solve the RPCA problem with samples that were not previously aligned. It uses a similar approach as TILT and works with batch alignments of linearly correlated images, instead of aligning single images.

Let $D = [I_1|\dots|I_n]$ be a matrix of observations where each column is an image stacked into a flat vector. By general assumption, its samples are not aligned and are possibly corrupted. Let $\tau = [\tau_1|\dots|\tau_n]$ be a set of domain transformations that act on each sample of D, in such a way that

$$D \circ \tau = [I_1 \circ \tau_1|\dots|I_n \circ \tau_n], \tag{2.12}$$

where resulting $I_j \circ \tau_j$ vectors may have a different dimension from the I_j ones. The entries of $D \circ \tau$ can be thought as selected regions of the entries of D that were

(a) Original images (b) Aligned images (c) Low-rank component

(d) Recovered errors (e) Average of (a), (b), (c), respectively

Figure 2.9: Example use of RASL to batch-align several faces of an individual under different illuminations, poses, expressions and occluding objects. The algorithm automatically finds a set of transformations τ such that the transformed images $D \circ \tau$ in (b) can be decomposed as the sum of a low-rank approximation A shown in (c) and a sparse component E shown in (d). The sharpened average faces in (e) demonstrate the how the alignment enhances the correlation between the transformed samples and even more in the low-rank term. Figure extracted from [46, 47].

geometric transformed by each entry of τ. Suppose A is a matrix that, given a proper τ, contains the aligned entries of D. In this sense, A is approximately low-rank and one can write $D \circ \tau = A + E$, where E incapsulates any possible corruption. With this constraint, RASL can be defined as the optimization problem:

$$\min_{A,E,\tau} \operatorname{rank}(A) + \gamma \|E\|_0 \quad \text{subject to} \quad D \circ \tau = A + E. \tag{2.13}$$

It is quite reasonable to assume that best alignment of the samples in D will minimize the rank of A. Assuming the sparsity of the E, the solution of problem (2.13) tries to simultaneously align and model the samples while compensating for the presence of any sparse corruptions. As TILT, problem (2.13) can be solved by usual convex relaxation and the local linearization of τ in each interation step [46, 47].

Figure 2.9 shows an example of RASL being used to batch-align several faces of an individual under different illuminations, poses, expressions and occluding objects. Images (a), (b), (c) and (d) in Figure 2.9 respectively show D, $D \circ \tau$, A and E. The average faces in (e) show the efficacy of the alignment, enhancing correlation between samples and consequently generating a more precise and compact linear model.

Some recent approaches, namely [51–53], explore the use of domain transformations to cope with misalignment between temporally close frames in background subtraction applications. Although the results of these methods are encouraging, none of them explores the use of videos acquired by moving cameras. Due to the nature of the problem, change detection in moving-camera videos demands the ability of comparing frames acquired at different times and whose field-of-view (FoV) only partially overlap. Thus, the use of domain transformations in this application requires this tool to compensate for much more than slight frame misalignments due to a camera jitter, as addressed by these previous publications.

Chapter 3

The Moving-Camera Surveillance Problem

This chapter introduces the literature of mobile camera solutions applied for anomaly detection in surveillance scenarios. Section 3.1 draws the basic setup of the problem in details, while Section 3.2 performs a short review of the state-of-the-art systems proposed to solve it. Finally, the VDAO video dataset is introduced in Section 3.3 as an important validation tool for mounted moving-camera experiments.

3.1 Problem Setup

Despite the success of several well established surveillance techniques using fixed cameras, the use of this type of solutions can be expensive or unpractical in certain complex scenarios. Fortunately, attaching a camera to a moving platform poses as an interesting work around to reach several viewpoints without increasing the number of cameras and, consequently, all the computational complexity related to them. This investigation addresses the problem of detecting changes in video sequences acquired by this kind of recording arrangements.

As seen in Chapter 2, linear approximation methods can be successfully employed on video surveillance problems to model video background with one or many low-rank subspaces and treating foreground anomalies as sparse outliers. However, dealing with backgrounds acquired with a moving camera can be significantly difficult, specially because static background structures may exhibit different speeds with respect to the camera, depending how far they are from the camera's position. Due to the perspective effect, structures closer to the camera will appear to move faster on the video. This phenomenon can break the linearity assumption, "confusing" the modeling process and introducing undesired sparse non-linear artifacts in the corruption component.

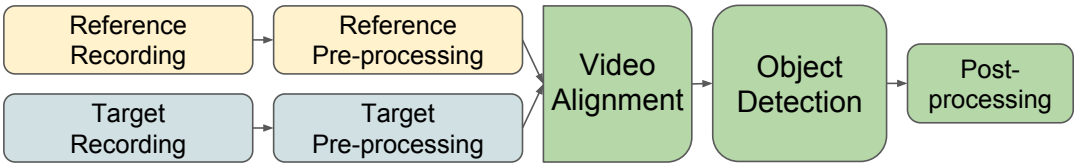


Figure 3.1: The generic framework of anomaly detection with a moving camera.

Another important issue is the way how modeling and detection are performed in the framework. For a static background, these two steps can easily be done simultaneously once a single-frame contains information of the whole background and anomalies are not expected to be present in every sample, or at least not in the same place. In the case of a moving camera's background, several samples may be needed for representing each point-of-view. Depending on the ammount and spatial resolution of the considered frames, bundle processing might be prohibitive. Additionally, the presence of outliers can induce the incorrect modeling of these objects as background if their relative frequency is high (like the standing man in Figure 2.1). In order to take into account all these concerns and enable the possibility of human validation of the samples that will accurately model the moving background, the framework proposed in this work will split modeling and detection into two different steps.

To describe the precise setup of the problem, some terminology is needed. Operator-validated sequences containing no anomalies are labeled as *reference videos*. These videos should be used to model the expected behaviour of the surveilled area. On the other hand, unsupervised video sequences that should be inquired by the detection algorithm in order to locate any abnormal presence are called *target videos*. Since most moving-camera techniques perform a frame-by-frame comparison, temporal and geometric alignment between reference and target videos is often required. Image processing techniques may also be considered to reduce the effect of undesired artifacts (e.g. noise, sporadic bright spots, illumination normalization, etc.). Figure 3.1 summarizes the usual framework that is followed by most of the solutions.

3.2 Moving-Camera Surveillance Systems

A few recent works try to solve the moving-camera detection problem by a diverse range of techniques. Solutions like [4], [6] rely on feature-tracking to perform foreground/background separation. However, since the videos are not structured in a reference-target manner, they are not suitable for comparison with our method.

One of the first attempts in anomaly-detection with moving cameras was done by trying to solve the problem of finding abandoned objects on the streets with a cam-

era mounted on a car, as proposed in [5]. The authors make use of GPS positioning to perform video alignment on the reference and target videos. To geometrically align corresponding target and reference frames, Scale-Invariant Feature Transform (SIFT) descriptors are obtained and compared using the Random Sampling Consensus (RANSAC) algorithm [48]. At last, the registered frames are compared by computing the Normalized Cross-Correlation (NCC) between the reference and target frames. The proposed method shows good performance results, but the need for external cues to align reference and target videos limits its applicability.

The approach presented in [31] works in a similar approach to [5] and is able to detect abandoned objects in heavily cluttered environments. Instead of making use of external cues to synchronize target and reference frames, it takes advantage of the camera's linear back-and-forth trajectory. Although it presents good performance and real-time results, the setup is somewhat dependent on the anomaly size, as well as the requirement of a specific type of camera movement to perform the video synchronization limits the algorithm usefulness in the case of a more general surveillance scenario.

The method described in [37] presents a dictionary learning with a two-stage strategy, with the advantage that it does not require the use of motion estimation or background subtraction. The dictionary is constructed to model portions of the reference video, while anomalies are detected as portions that are not well represented. This approach also dispenses the use video synchronization and geometric registration. However, the dictionary construction time is high and the results show a high rate of false alarms.

More recently, the method proposed in [36] describes a new approach for anomaly detection in moving-camera video signals based on a sparse representation of both the reference and target sequences. In the proposed system, the reference video is represented as the combination of a low-rank projection onto a union of subspaces and a sparse residue [32], which are then employed to represent the target video. The residue of this last representation allows one to identify video anomalies in the target video. This scheme obviates the need of temporal alignment between the two video sequences.

In this work, we propose a new method to detect changes in moving-camera video sequences by performing a sparse representation of the target video in a transformed domain that copes with high levels of geometric misalignment with the reference video.

3.3 Testing Datasets

In order to efficiently verify the detection quality of our proposed technique, an appropriate set of video sequences that fits the moving-camera setup was chosen, along with its corresponding *ground truth* information. Very popular testing datasets extensively used by similar works are the image sequences introduced in [9, 10]. Both of them present image sequences acquired with a static camera, generally taken at public spaces with people moving around, sometimes with varying illumination or dynamic background. Unfortunately, none of these datasets present sequences with a moving camera. The videos presented in [11] were provided as a benchmark for testing feature-based motion segmentation algorithms and contain a few images with a moving camera, but they were not taken minding camera's periodic pose repeatability and do not fit the needs of this work. A more recent dataset described in [12] presents several situations divided into challenge categories, like "dynamic background", "camera jitter" and "intermitent object motion". In 2014 [13], new categories were introduced in a extended version of the dataset, with a particular interesting "PTZ"[1] category, that displayed actual moving backgrounds in a periodic fashion. However, the training data is not clean of outliers and the technique should be adapted to correctly model the whole background. Furthermore, fixed PTZ cameras only perform changes in pose, not actually changing the camera point-of-view.

3.3.1 VDAO: an Abandoned Object Database

At the time of writing, the VDAO dataset [14] was the only publicly available dataset designed for object-detection in moving-camera video sequences, to the best of our knowledge [17]. This database contains several recordings of abandoned objects on an industrial-like facility, simulating a real-world scenario with great complexity and barelly controlled illumination. The dataset sequences were acquired with a rigid camera mounted on a robotic iRobot™ *Roomba* platform with a back-and-forth linear movement on a 6m-long hanging rail. Both track and robot can be seen in Figure 3.2. Two different IP cameras were employed, having the same 1280×720 pixel resolution and frame rate of 24 fps. Twenty four distinct abandoned objects were employed in the recordings, which total approximately 8.2 hours of video. The recordings were acquired at a complex scenario, comprised of several pipes, valves and other visually complex structures, posing as a greater challenge compared to usual databases. In VDAO, the recordings were divided into two groups: reference and target sequences, as exemplified in Figure 1.2. The reference sequences have no abandoned objects, as validated by human supervision, while the target sequences

[1] PTZ: abbreviation for pan-tilt-zoom.

Figure 3.2: Pictures taken at the facility where the VDAO sequences were recorded. The left picture shows and overview of the industrial environment and the hanging rail. The right one exhibits a close shot of the iRobot™ Roomba on the rail track with the mounted camera.

contain one or more objects to be detected automatically by the proposed algorithm. Due to track imperfections and mechanical friction with the robot wheels, the captured sequences present considerable camera shake, which poses an additional challenge to the detection scheme.

In VDAO, the positions of all abandoned objects along the target videos are identified, frame-by-frame, in a separate file by their corresponding bounding boxes, making it possible for any detection method to be rigorously assessed. A simple annotation system was developed to mark up the ground truth data available in the database. A sample session of this software is displayed in Figure 3.3. Object positions can be marked with rectangular bouding boxes for every frame in all target sequences. For objects appearing in multiple connected components, subobject annotations are also provided, as can be seen in the backpack occluded by a pipe in Figure 3.3. This database, along with the ground truth annotations of the abandoned objects, can be downloaded from [15]. The complete details of the database construction can be seen in [14].

3.3.2 VDAO-200

Since VDAO contains massive video files, which can be prohibitive in terms of the average memory and processing power of current computers, dealing with a smaller and representative sample of the database is an interesting option. To this end, a special selection of the database, called VDAO-200 [16], was developed in order to standardize a simple subset for detection quality assessing and future comparisons between methods.

This auxiliary database is composed of 59 excerpts with 200 frames taken from VDAO single-object target videos. The selection contains a total of 9 different ob-

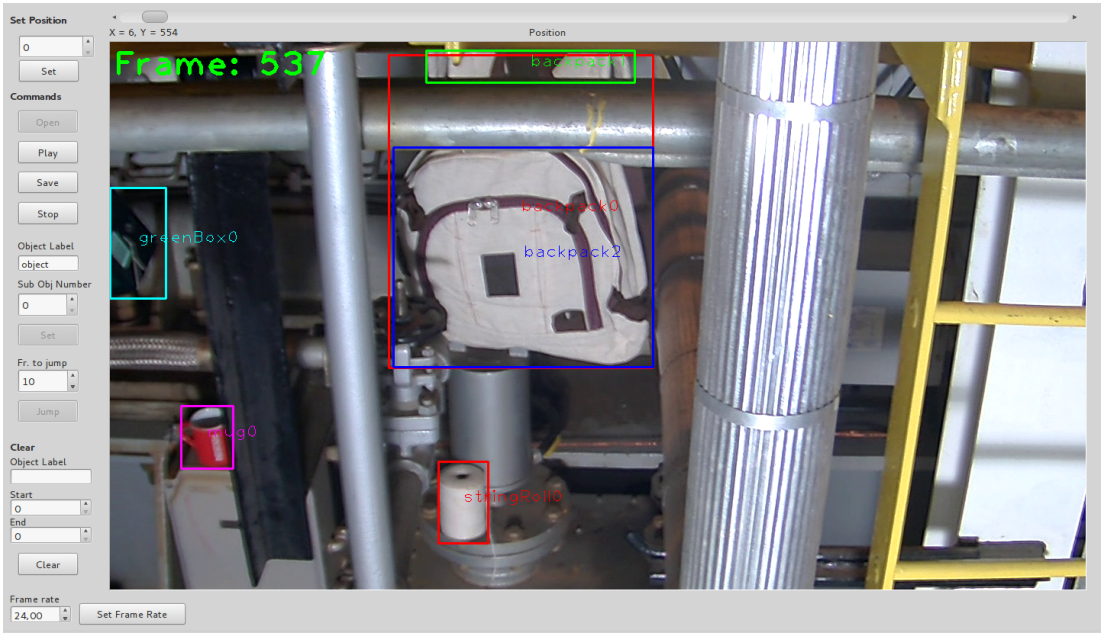


Figure 3.3: Sample session with the ground truth annotation system for the VDAO database. Rectangular bounding boxes determine the approximate ground truth position of the abandoned objects in target videos [14, 15].

jects in different positions and 2 types of illumination. On almost half of the videos, the objects are partially or completely occluded. There are also several situations of environment shadow casting, different object shapes and camera shaking, which make the selection very challenging and also representative of the full database.

Chapter 4

Moving-Camera Surveillance with Low-Rank Representation

A first solution using principal subspace analysis is proposed in this chapter in order to solve the moving-camera problem. Section 4.1 shows how to adapt the RoSuRe technique (seen in Section 2.1.1) to the problem at hand and some preliminary results can be seen Section 4.2. Section 4.3 describes a post-processing stage that converts the output of the proposed algorithm to a binary video that represents the locations of the detected anomalies. Finally, Section 4.4 describes a methodology to tune the solution's parameters to the testing data and the cross-validated results of this essay. This methodology is applied to an excerpt of the VDAO dataset, as explained in Section 4.4.1.

4.1 Adaptating RoSuRe to handle the Moving-Camera Setup

Let X_r be a data matrix whose columns are composed by reference samples, generally the frames from a reference video stacked as long column vectors. The modeling step uses the RoSuRe algorithm to decompose X_r according to Eq. (2.6) that is,

$$X_r = L_r W_r + E_r, \tag{4.1}$$

where L_r is the blockwise low-rank part of the reference samples and is assumed to resemble the sampling of a union of linear subspaces, whose structure is described in W_r. For this model, E_r is the non-linear component, which is assumed to be sparse. Notice that E_r is not seen as matrix of anomalies, but residual information that could not fit the recovered model in L_r.

Now let X_t be a matrix of target samples, analogous to X_r. Assuming that X_t

shares the same subspace structure with its reference counterpart, one can rewrite the blockwise low-rank part of X_t as a combination of the columns of L_r plus a sparse residual. In other words, one can find sparse matrices W_t and E_t, such that the target data matrix can be written as

$$X_t = L_r W_t + E_t. \tag{4.2}$$

Using this description, all anomalies in X_t, such as an abandoned object or and unauthorized passerby are encapsulated into E_t. In order to perform this alternate representation of X_t, taking advantage of L_r determined in the decomposition (4.1), the RoSuRe algorithm must be modified to work with a given fixed term L_r. To do so, the cost function in Eq. (2.7) should be modified to

$$\min_{E,W} = ||W||_1 + \lambda||E||_1 \quad \text{s.t.} \quad L_r W = X - E, \tag{4.3}$$

and, following Eq. (2.8), the new augmented Lagrangian function becomes

$$\hat{\mathcal{L}}(W, E, Y, \mu) = ||W||_1 + \lambda||E||_1 + \underbrace{\langle L_r W - X + E, Y\rangle + \frac{\mu}{2}||L_r W - X + E||^2}_{g(W,E)}. \tag{4.4}$$

In a similar way to $f(W, E)$ in Eq. (2.8), $g(W, E)$ is the smooth part of the Lagrangian and will be used to compute the update steps of W_k and E_k as follows:

$$\begin{aligned}
W_{k+1} &= \arg\min_{W} ||W||_1 + g(W, E) \\
&= \tau_{\frac{1}{\mu\eta_1}}\left[W_k - \frac{1}{\eta_1}\nabla_W g(W_k, E_k)\right] \\
&= \tau_{\frac{1}{\mu\eta_1}}\left[W_k - \frac{1}{\eta_1}{L_r}^T\left(L_r W_k - X + E_k + \frac{Y_k}{\mu_k}\right)\right], \qquad (4.5) \\
E_{k+1} &= \arg\min_{E} \lambda||E||_1 + g(W, E) \\
&= \tau_{\frac{\lambda}{\mu\eta_2}}\left[E_k - \frac{1}{\eta_2}\nabla_E g(W_{k+1}, E_k)\right] \\
&= \tau_{\frac{\lambda}{\mu\eta_2}}\left[E_k - \frac{1}{\eta_2}\left(L_r W_{k+1} - X + E_k + \frac{Y_k}{\mu}\right)\right], \qquad (4.6)
\end{aligned}$$

as summarized in Algorithm 2. As mentioned before, performing this procedure in Eq. (4.2) tends to isolate in E_t all the target-sample information that is not present in L_r (or X_r). However, besides the sparse corruptions generated by the anomalies, E_t will also have the residual sparse non-linear information that could not be captured by the blockwise low-rank $L_r W_t$ representation. The outlier information contained in E_t can be separated from its inherent non-linear residual by noting that, as

Algorithm 2 - Sparse representation of X given blockwise low-rank L.

Input: $L, X, \lambda, \rho > 1, \eta_1, \eta_2$
while not converged **do**
$L'_{k+1} = X - E_k$
$W_{k+1} = \tau_{\frac{1}{\mu\eta_1}} \left[W_k - \frac{1}{\eta_1} L^T \left(LW_k - L'_{k+1} + \frac{Y_k}{\mu_k} \right) \right]$
$E_{k+1} = \tau_{\frac{\lambda}{\mu\eta_2}} \left[E_k - \frac{1}{\eta_2} \left(LW_{k+1} - L'_{k+1} + \frac{Y_k}{\mu_k} \right) \right]$
$Y_{k+1} = Y_k + \mu_k \left(LW_{k+1} - L'_{k+1} \right)$
$\mu_{k+1} = \rho\mu_k$
end

X_t and X_r are similar by assumption, E_t will look in general quite similar to E_r, except around these anomalies. Therefore, we also perform a third and last step, decomposing E_t using E_r as the input parameter L of Algorithm 2, yielding

$$E_t = E_r W + E. \tag{4.7}$$

The remaining sparse component E tends to contain, as desired, just the outliers in X_t not present in X_r. To corroborate the efficiency of this framework, target videos with abandoned objects and their corresponding reference counterparts are used as input and the results are presented in the next section. Figure 4.1 summarizes the mcRoSuRe method optimization workflow.

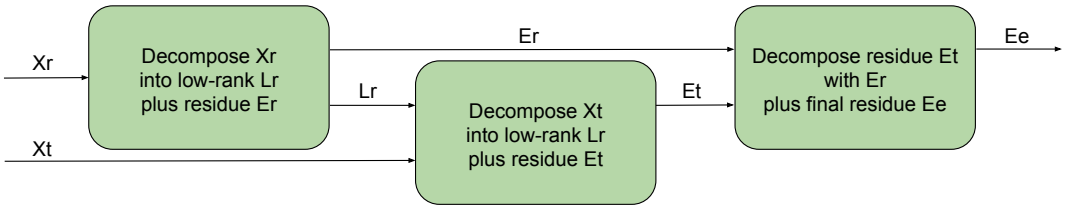


Figure 4.1: the mcRoSuRe optimization workflow.

4.2 Preliminary Results

The proposed strategy was tested for several video sequences from the VDAO database [14] that is available from [15]. As cited before, in order to ensure a proper decomposition of the target video sequence X_t using the blockwise low-rank approximation L_r of the reference video sequence X_r, we have to guarantee that all frames of X_t have sufficient corresponding frame samples in the reference video X_r. Our experiments have shown that reference and target recordings of the same surveilled area with equal sampling densities are sufficient to exhibit impressive re-

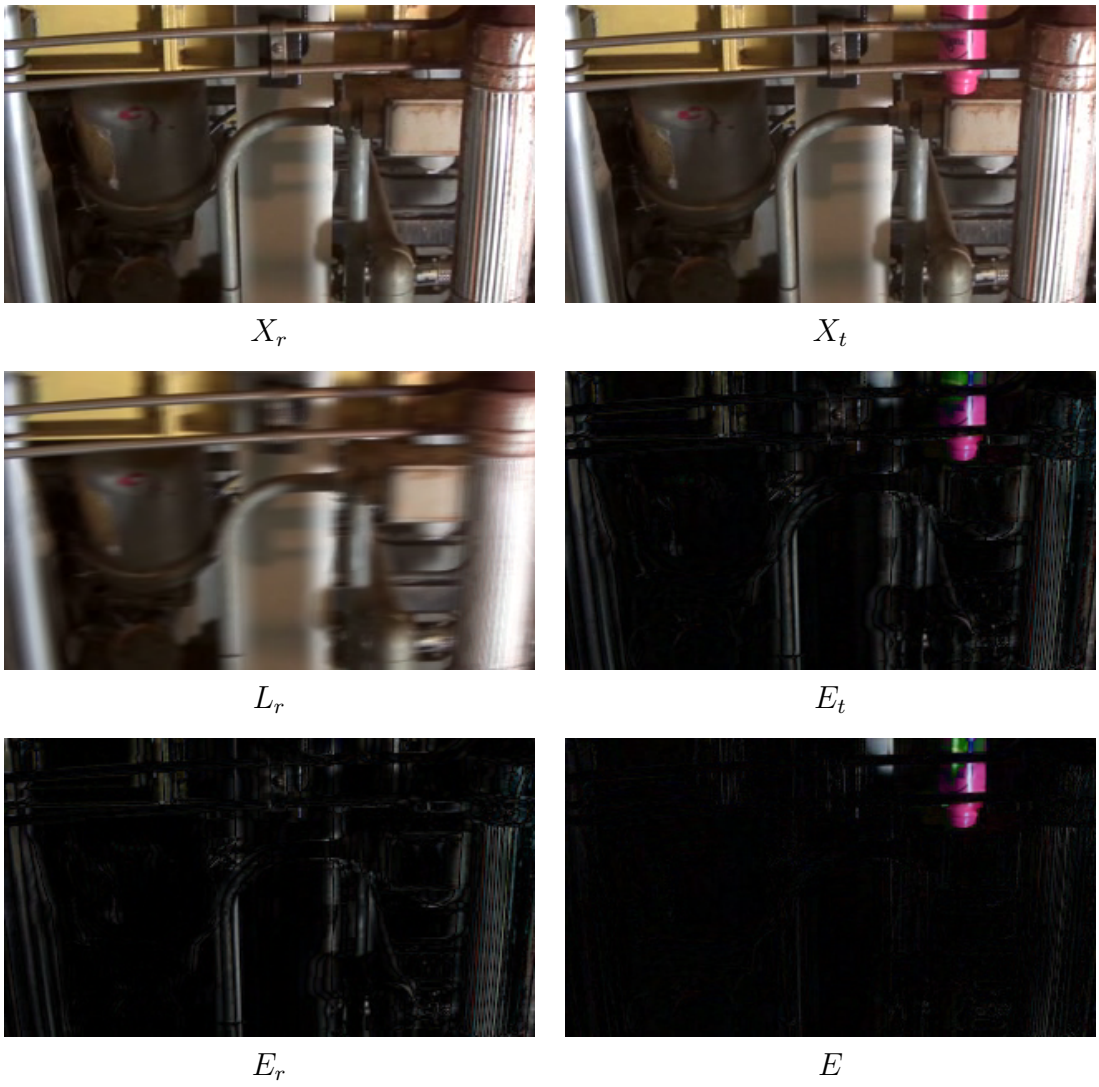


Figure 4.2: Experimental results (single frames of matrices X_r, X_t, L_r, E_r, E_t, and E) using proposed representation for abandoned-object scenario (pink bottle).

sults. Since both types of video cover the same area and were acquired with the same velocity, the sampling conditions are met if X_r contains at least one complete lap the camera through the surveillance area.

Unfortunately, there are computational complexity limitations and a complete surveillance lap of the reference video cannot be processed on a regular computer. In this case, one can follow a divide-and-conquer approach by segmenting both the reference and target videos with the care that each segment of the reference sequence contains all the corresponding frames of the target sequence. Moreover, video sequences were spatially subsampled to 320×180 pixels.

Results for four different abandoned-object situations are illustrated in Figures 4.2,4.3, 4.4 and 4.5. In each case, a 70-frame reference sequence is used to model a 50-frame target sequence contained within the reference sequence. In all steps, algorithms 1 and 2 employed $\lambda = 1$ and $\rho = 1.5$. In all figures, each image

Figure 4.3: Experimental results (single frames of matrices X_r, X_t, L_r, E_r, E_t, and E) using proposed representation for abandoned-object scenario (backpack + wrench + box).

represent a sample frame of the matrices described in Equations (4.1), (4.2), and (4.7). More precisely, X_r, X_t, L_r, E_r, E_t, and E. Each figure represents a given experiment. The sparse component matrices E_r, E_t and E are visualized in terms of their absolute values, since these matrices may contain negatives values. The E matrix contains the detected abandoned objects and any other differences detected in the respective target frame labeled as X_t. This procedure is performed on each of the three RGB channels and the results are combined into a single colored image.

One can see that the proposed method can detect the abandoned objects while having very few false positives. Local illumination changes due to shadow casting, as well as major camera misalignments, may also lead to false positives. Strong shadow casting can be clearly noticed at the left of the pink bottle (Figure 4.2) and at the left of the white bottle (Figure 4.5). Subtle camera misalignments artifacts

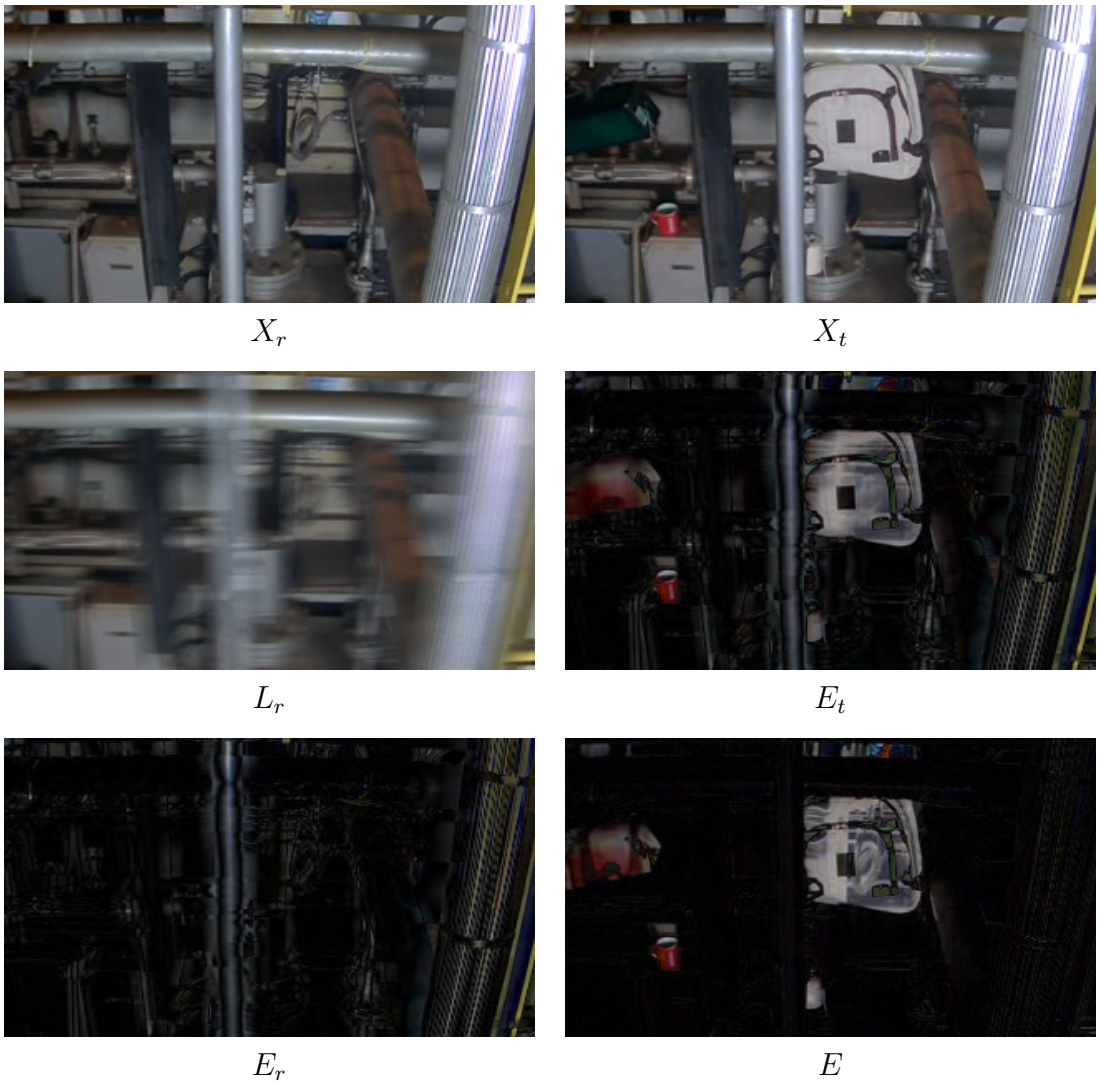


Figure 4.4: Experimental results (single frames of matrices X_r, X_t, L_r, E_r, E_t, and E) using proposed representation for abandoned-object scenario (backpack + green box + mug + string roll).

can be seen at the right of the backpack in Figure 4.4. False negatives may occur if the abandoned object has the same pixel intensities as the background, as seen in the middle of the wrench in Figure 4.3. Most of these artifacts, however, can be removed by simple image-processing techniques, as proposed in the next section.

4.3 Post-Processing

As seen in the last section, the E matrix obtainted in the decomposition step described in (4.7) yields an approximated *background subtracted* version of the target sequence X_t. Therefore, pixels in anomalous regions of E are likely to have higher intensities in absolute value, while pixel intensities outside these regions are expected to be close to zero. Since the inquired database comes along with annotated

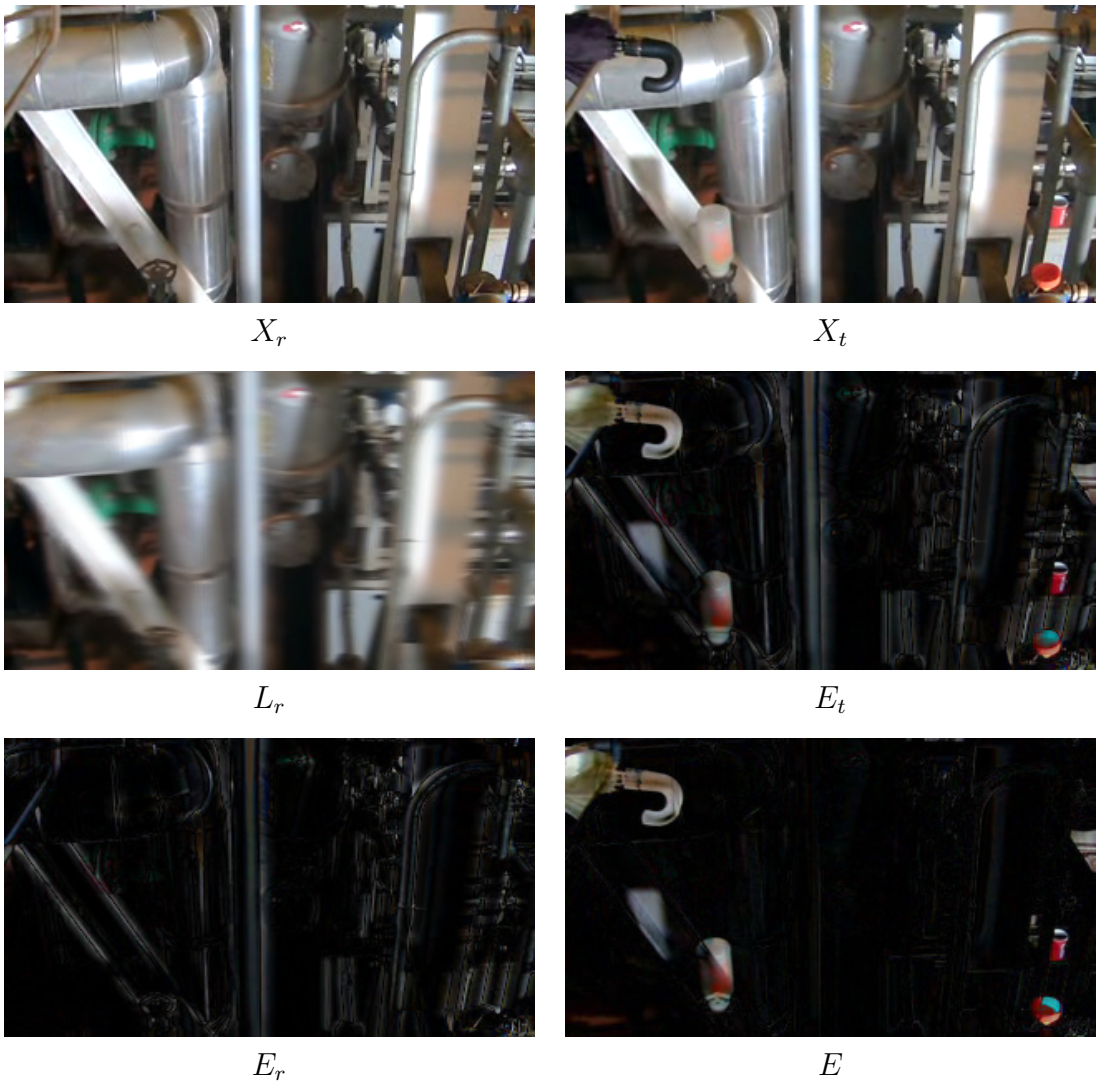


Figure 4.5: Experimental results (single frames of matrices X_r, X_t, L_r, E_r, E_t, and E) using proposed representation for abandoned-object scenario (umbrella + bottle + bottle cap + mug).

information about the placement of all abandoned objects that appear in the target sequences, detection quality can be assessed by verifying if the revealed objects in the frames of E lie inside the *bounding rectangles* that indicate the individual presence of these objects. To evaluate the robustness of the proposed technique and perform a fine adjustment of its free parameters, the frames of E must be binarized and pixelwise compared with the annotated bounding rectangles, by assigning pixel value to 1 in the presence of anomalies, and, otherwise, set to 0. The parameter setup as well as the used metrics are described in the next chapter.

The post processing stage consists of a simple binary thresholding with four additional steps, comprising filtering and morphological operations. These steps were chosen to maximize the accuracy of the detection, eliminating major false alarms. The first step is applied before binary thresholding and consists of an

average spatial filtering of each unpacked frame of E to reduce noise and scatter pixel energy, exploiting the presumed continuous nature of E. Let E_0 be E reshaped as a three-dimensional tensor representing a sequence of images in the classic MATLAB fashion (height $\times$ width $\times$ frame). An integer parameter ω is used to represent the lenght of the edge of the filter's square kernel, which means that it is a $\omega \times \omega$ window. Let E_1 be the resulting tensor after filtering E_0. Binary thresholding occurs after this first step and each pixel of E_1 is converted to 1, if its absolute value is grater than a β threshold, otherwise is set to zero and let E_2 be the resulting tensor after this thresholding.

Then, two morphological operations are applied in sequence. First, a *morphological open* with a square-shaped structure element is applied in order to remove small spurious detections, followed by a *morphological close* with square-shaped structure element used to join possible disconnected regions of the same detect object, since it is possible that part of the anomalies might share similar pixel intensities with its corresponding reference background. Let μ_1 and μ_2 be the respective (integer) lenghts of the edges of the open and close structure elements, and let E_3 and E_4 be the resulting tensors after open and close operations, respectively. Finally, a temporal voting along each pixel is performed in order to cull incorrect detections caused by severe misalignments between reference and target sequences. This kind of artifact tends to occur locally in time, opposed to anomalies like abandoned objects that continously persists on some pixel neighborhood for some time. This assumption is suitable for a temporal voting process in order to eliminate sporadic detections and preserve persistent ones. The voting is done by using a odd-sized window of lenght $2\kappa + 1$ centered at each pixel of E_4, along time-dimension, where κ is an integer paramater. If at least half of the pixels are set to 1, the resulting pixel will also set to 1, otherwise it is set to 0. Let B be the resulting tensor after applying this voting on E_4. B is also the final tensor generated in the pre-processing stage and is the input of the evaluation metrics, and is completely summarizes in Figure 4.6.

4.4 Algorithm Setup and Parameter Adjustment

The proposed method contains a total of six parameters (the optimization one and the five other described in the last section) and in this section we suggest a tuning scheme based on a hyperparametric search basis. Due to pratical hardware and time limitations, a subset of VDAO database was selected, subsampled and sliced into short videos (i.e., few number of frames), before they were actually used in the experiment, as we will describe in the Section 4.4.1. The parametric tuning is presented in the following section describing a rigorous and extensive experiment realized by employing a K-fold cross validation scheme. The results are exhibited

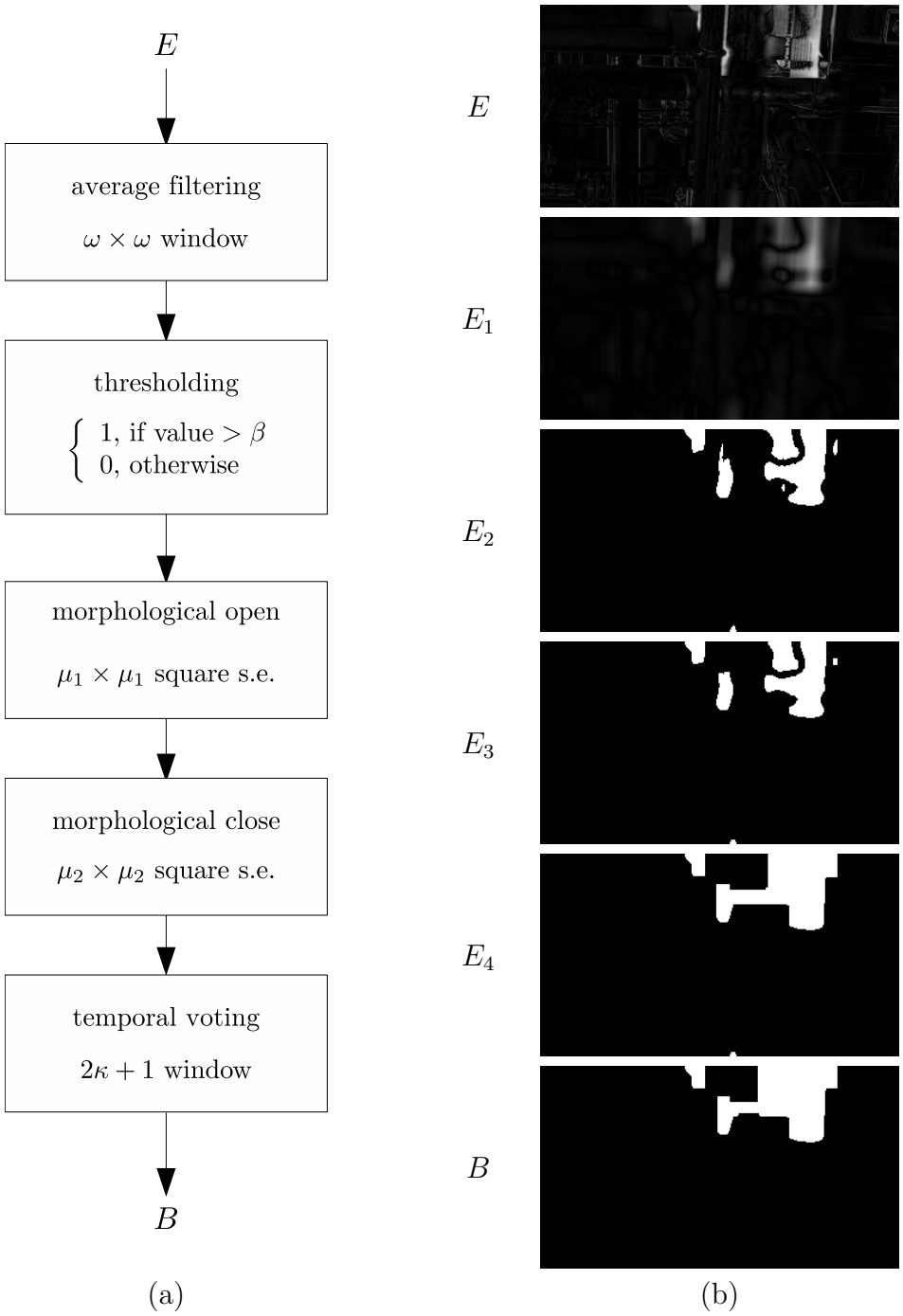


Figure 4.6: Post-processing steps: (a) shows a diagram of the summarizing the post-processing step and (b) shows a sample frame at each step.

in the last chapter together with commentaries.

4.4.1 Batch of Video Slices

Each video in the VDAO database contains at least one full lap of the robot along the guiding track, which takes about six minutes of recorded video with 1280 × 720 pixel resolution at 24 frames per seconds, summing up a total of 8 billion pixels. Considering that this data will be used in matricial operations, a single uncompressed color

video should ocupy about 96 GB of RAM, if values are modestly packed into 32-bit floating-point format. Such amount of memory bears the edge of current computer capabilities, becoming impratical to use full videos in the following experiments.

To deal with this, a simple and effective alternative is to work with short, randomly selected video slices, that conform to a small set of constraints to ensure the testing batch is sufficiently representative. As mentioned before, VDAO database has single- and multi-object videos sets to allow parameter adjustment and model validation for different object sizes. For this experiment, only single-object videos with extra illumination were considered. A subset of 12 target videos and their respective reference videos were chosen to be used.

The slicing constraints take into account the prevalence of selected frames that contain abandoned objects as an attempt to equally distribute its presence between frames for each target video, chosen to be between 45 and 55 percent. A minimal gap of 200 frames between slices is also imposed to avoid slice overlapping. From each target video, 4 video slices were spanned, totalizing 48 target slices. Only the first half-lap of each target video is considered, to avoid slice overlapping. Assuming constant speed of the platform robot, an equal number of frames approximately covers the same visual range in different slices if they start in the same position and direction. Considering this, for each target slice, a brute-force frame-comparing algorithm searches for the best corresponding reference slice with the same number of frames in a pixel-by-pixel, absolute difference basis. At the end, the slice is manually validated.

Since RoSuRe decomposition exploits frame similarities, each frame in target slice should be sufficiently represented by its reference slice. As insurance, after a target and reference slice match, reference slice is expanded both ways (before and after) by p padding frames at each side. That means, if a target slice of size N_t matches a reference slice of the same size at position n, the first reference slice frame will be $n - p$, the last one will be $n + N_t + p - 1$ and the number of frames in reference slice will be $N_r = N_t + 2p$. The chosen values for this experiment were $N_t = 50$, $p = 10$ and $N_r = 70$.

Finally, frames were appropriately subsampled to 320×180 pixels and converted to grayscale using a classic luminance formula[1]. Ground truth annotation data were also converted to new pixel resolution and frame range, for each slice. No frame decimation was performed. Minding experiment reproducibility, all described tasks for video slicing and ground truth conversion were coded into an utility command-line script. This software is freely available and may be used for future comparisons.

[1] $Y = 0.299R + 0.587G + 0.114B$

4.4.2 Parameters Adjustment

As seen in the last chapter, the complete framework ended up with six tunable parameters: one (λ) in the matrix decomposition stage and five (β, ω, μ_1, μ_2 and κ) in the post-processing stage. Since λ is a positive real value representing the optimization weight in (2.7) and (4.4), the minimization of $||W||_1$ will be prioritized if $\lambda \in (0, 1)$. Conversely, $\lambda \in (1, +\infty)$ will give priority to minimize $||E||_1$. It is easy to see that inverse values of λ will give complementary importance to $||W||_1$ or $||E||_1$, depending on each case. In this sense, symetric values of λ are not equally distributed over the search range. Instead, fitting the exponent $\gamma = \log_2 \lambda$ solves this issue, once γ_0 and $-\gamma_0$ are symetric in the search range and $\lambda_0 = 2^{\gamma_0}$ and $\lambda_0^{-1} = 2^{-\gamma_0}$ have complementary importance, giving equal oportunities to symetric values on the adjustment.

Although not many, six parameters might be hard enough to perform an exaustive search due to the combinatorial growth of tests. If each parameter dimension were divided into 10 possible values and considering that the average time to proccess a single video slice is about 20 seconds, a full search will require about 231 days to process one video per fold. Alternatively, assuming some continuity of each evaluated property along a six-dimensional parameter range, a recursive search can test fewer parameter combinations and still achieve a local maximum with fewer levels of recursion.

To deal with integer values, discrete parameters (ω, μ_1, μ_2 and κ) are treated as real values for the recursive search, but are conveniently truncated to integer values before testing. For example, consider a hypothetical parameter that varies from 1 to n. To give equal probability for each integer value, the corresponding real range must be $[1, n + 1]$, which contains n unit-spaced ranges. For this experiment $\gamma \in [-9, 9]$, $\beta \in [0, 1]$, $\omega \in [1, 31]$, $\mu_1 \in [0, 31]$, $\mu_2 \in [0, 51]$ and $\kappa \in [0, 11]$, reminding that truncated values of the discrete parameters will only reach the upper limit minus one, (e.g. $\lfloor \kappa \rfloor \in \{0, 1, \ldots, 10\}$ and so on).

4.4.3 Recursive Subdivision Search

In order to avoid excessive testing of parameter combinations, a recursive subdivision strategy was used to estimate the best parameter set for each testing fold. The subdivision scheme is inspired by classic binary subdivision, shortening the search range at each step. Let $\pi^1, \pi^2, \ldots, \pi^p$ be the fitting parameters and let π_a^j and π_b^j respectively be the lower and upper bounds of parameter π^j, such that $\pi^j \in [\pi_a^j, \pi_b^j]$, for each $j \in 1, \ldots, p$. At each recursion level, for each parameter π^j, the mean values of each half of $[\pi_a^j, \pi_b^j]$, whose values are $\pi_a^j + s^j$ and $\pi_a^j + 3s^j$, where $s^j = (\pi_b^j - \pi_a^j)/4$, are combined to generate a set of 2^p points that are uniformely distributed inside

the search volume $[\pi_a^1, \pi_b^1] \times \cdots \times [\pi_a^p, \pi_b^p]$. At each level l, all 2^p points are tested against the assessing metric and the point with the best score is also compared with the best point of the prior level $(l-1)$, which happens to be the center of the search volume at each level. Thus, the resulting best point of level l is chosen to be the center of the range volume in level $l+1$, that is shortened to half in each dimension. This is easily done by assigning $\pi_a^j = q_l^j - s^j$ and $\pi_b^j = q_l^j + s^j$ at level $l+1$, where q_l^j is the j-coordinate of the best score point q_l in level l. Since there are no points before the first level, the score at the center of the initial range volume is used in place, for algorithmic completeness. To illustrate the subdivision process, see Figure 4.7.

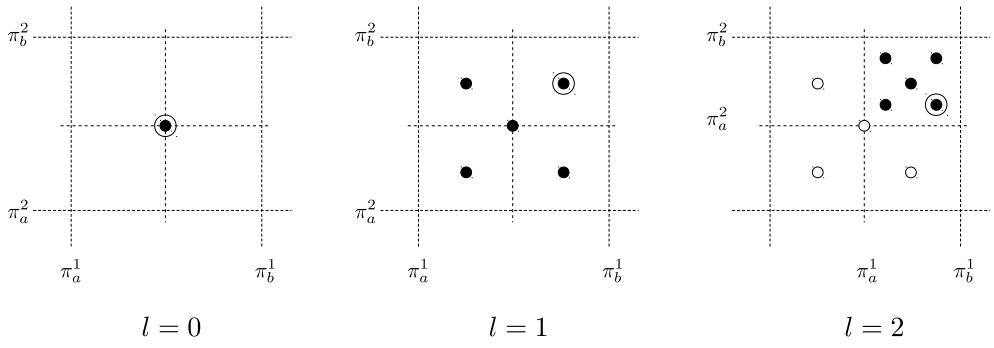


Figure 4.7: Example of recursive subdivision with 2 parameters and 2 levels. First (left), the score of the midpoint of the initial range is computed. At the first level (middle), 4 points are generated and the best score point is selected, considering the midpoint score. At level 2, the range volume is shortened by half at each parameter dimension, with center at the best point of previous level ($l = 1$). Circled dots highlight the best score points at each level.

Metrics

In classification literature, *recall* (also known as *sensitivity* or *true positive rate*) and *precision* (also called *positive predictive value*) play a major role in classifier evaluation. A good classification is expected to present high rates of *true positives*, which is measured by recall, and low rates of *false alarms*, indicated by the precision. Since these metrics are not complementary, maximizing of one them does not guarantee the maximization of the other. A commonly used overall metric is the *F-measure* (or *F1-measure*), that combines recall and precision by performing their harmonic mean. Thus, the F-measure was the chosen score as the parametric tunning objective function, as it promotes a conservative trade off between the two concerned metrics.

To compute the recall of a given tested slice, one should use the provided VDAO annotations which indicate the placement of the abandoned objects over the target videos. For each target frame, a binary image indicating the approximated support of the appearing abandoned object (if any) is generated. This binary image is used

as a logical mask to compute the correctly detected pixels on the corresponding frame of the binary output tensor B, as described in Section 4.4.2. Let M be a binary tensor collecting all these binary masks for some processed output tensor B, then the computation of recall (Re) can be easily done with simple logical operations and counting the nonzero pixels of the following binary tensors

$$Re = \frac{\sum_i (B \cap M)_i}{\sum_i M_i}, \tag{4.8}$$

where i subscript is the i - th entry of each tensor (seen as a flat vector).

Notice recall quantifies the percentage of correct detections (nonzero entries of the intesection of B and M) over all the expected detections (nonzero entries of M). Analogously, precision (Pr) can be defined as

$$Pr = \frac{\sum_i (B \cap M)_i}{\sum_i B_i}, \tag{4.9}$$

which quantifies the percentage of correct detections over all detections (nonzero entries of B). F-measure is simply computed by the formula

$$Fm = \frac{2RePr}{Re + Pr}. \tag{4.10}$$

4.4.4 Cross-Validation Results

For the K-fold validation, 12 target videos were used to span 4 slices each, totalizing 48 target slices. For each fold, slices were divided into *training* and *testing sets*. Each training set was then submitted to the procedure described in Section 4.4.2 to determine the best parameter adjustment. After that, these parameters were applyed to the testing set. Recall and precision were computed with Eqs. (4.8) and (4.9) by aggregating binary results of all slices of the concerned set into a single division (i.e. summing all correct detections of all slices in the set and dividing by the sum of all expected detections of all slices in the same set, for each training and testing sets in each fold). The results of this cross validation can be seen in Table 4.1. The second column of table shows the recall (Re), precision (Pr) and F-measure (Fm) scores of the best parameters adjusted for the training set in each fold. The third column shows the scores for the testing set with the same parameters, which in turn can be seen in the fourth column. The last row presents the mean and standard deviation of each score and parameter throughout the folds.

The target slices were arranged in such a way to avoid biasing by ensuring that slices extracted from the same target video cannot be at the training and testing sets at the same time in the same fold. This is simply done by grouping slices of

	training set			testing set			adjusted parameters					
fold	Re	Pr	Fm	Re	Pr	Fm	γ	β	ω	μ_1	μ_2	κ
1	0.52	0.60	0.56	0.36	0.34	0.35	6.750	0.125	12	12	45	1
2	0.54	0.57	0.55	0.19	0.24	0.22	6.750	0.125	12	12	45	1
3	0.41	0.39	0.40	0.61	0.93	0.74	7.313	0.156	11	5	30	1
4	0.57	0.41	0.48	0.92	0.50	0.65	6.750	0.125	12	4	32	1
5	0.60	0.44	0.51	0.50	0.21	0.29	6.750	0.125	12	4	32	1
6	0.56	0.46	0.50	0.56	0.37	0.45	7.313	0.094	13	13	43	1
μ	0.53	0.48	0.50	0.53	0.43	0.45	6.938	0.125	12.00	8.33	37.83	1.00
σ	0.05	0.07	0.04	0.17	0.19	0.16	0.250	0.010	0.33	4.00	6.50	0.00

Table 4.1: Results of K-fold cross validation with $K = 6$ and $F = 8$

the same target video and setting the *fold size* F (which is also the testing set size) to be a multiple of the number of slices spanned by each video (in this case, 4). To avoid overfitting and still have a reasonable number of folds, K was set to 6 and F set to 8, which means that each testing set contains 8 slices spanned from exactly 2 target videos and each traning set contains 40 slices spanned from exactly 10 target videos. With this conditioning, training and testing sets do not share slices from a same target video, for each one of the 6 folds.

Commenting the results

Before analysing the results, it is important to note that VDAO provides bounding rectangles as the annotated locations of the abandoned objects present in the target videos. Since we chose a pixelwise metric, even if some object is fully detected, *Re* will not achieve full score unless its shape resembles a perfect rectangle with sides parallel to the canonical axes. Therefore, it is important to realize that *Re* can vary with the object's visible shape along the same video and majorly between videos. This alone explains why the average *Re* scores are not very high. One might wonder the reason we do not use an objectwise or framewise metric to achieve more effective results. Indeed, pixelwise metric are smoother, making them more appropriate to perform the parametric regressions. It is also important to note that the some target videos in the VDAO database have severe pose misalignment between them and their corresponding reference videos. Unfortunately, it is quite notorious in literature that these kind of linear modeling techniques are likely to fail under these circunstances [28]. In practice, gross local misalignments will be treated as outliers, raising the rate of false alarms. Knowing this, results should be seen more qualitatively, once we are dealing with a very complex and open problem and this research is a work in progress.

In a first glance, the results displayed in Table 4.1 show very positive messages, but some caution remarks are also required. The mean scores of the testing sets remain very similar to the ones of the training set, meaning that the proposed tech-

nique exhibit good overall robustness, but it also present increased variance between the testing scores. The most plausible explanation for this discrepancy, apart from the one given above, is that the size of the training set is five times larger than the testing set, which might attenuate the standard deviation, aggravated by fact that, even containing 8 video slices, the testing set contains information from 2 target videos only. Another encouraging outcome is that most parameters appear to converge near one global limit, excecpt from the morphological operations, which appear to have two local optima. An apparent reason for this fact is that maximizing Fm can be achieved by increasing both Re and Pr. However, these metrics often compete with each other. For example, increasing Re implies increasing sensitivity which can lead to more false alarms. Conversely, a more precise algorithm should be more cautious, reducing sensitivity. Since some videos have serious pose misalignments, the parameter regression algorithm might choose different morphological operations to cope with the resulting false alarms.

To illustrate this problem, two different sample cases were selected: one with good reference alignment and the other with severe misalignments. Parameters were set to the mean values of table 4.1, that is $\gamma = 6.938$, $\beta = 0.125$, $\omega = 12$, $\kappa = 1$, excecpt from μ_1 and μ_2 that were assigned with the two likely optima indicated in the table. Let B_1 be the resulting binary image for first setup where $\mu_1 = 4$ and $\mu_2 = 32$ and B_2 the resulting image for $\mu_1 = 12$ and $\mu_2 = 45$. Both setups were applied to both cases and the results sample frames can be seen in Figure 4.8. The first and second rows show reference and target frame samples for each case. The third row shows resulting binary image for the $B1$ ($\mu_1 = 4, \mu_2 = 32$) setup and the forth row show the result for the $B2$ ($\mu_1 = 12, \mu_2 = 45$).

Column (a) in Figure 4.8 shows the results for a well aligned case. Both post-processing setups exhibit good detection of the abandoned box at the top left of the scene, with practicaly no false alarms. Also notice the B_2 setup peforms slightly better due to a wider structure element of the morphological close (μ_2), increasing the Re score. On the other hand, the results shown in column (b) present gross misalignments between reference and target sequences, producing several false alarms. In the B_1 setup, the dark shoe at the left of the scene was partially spotted with very low precision, due to the many erroneous detections. B_2 setup removes a good load of false alarms, but unfortunately it could not detect the shoe. This kind of traid-off seems to explain why the algorithm has two optimal adjustments. However, post-processing is not an effective solution for poor modeling.

Apart from the severe misalignment cases similar to one seen in column (b), the results of the proposed method are quite impressive, considering the significant amount of camera shaking in all VDAO videos and the fact it does not need any kind of image registration. This framework should perform very well in cases of a

smooth camera movement and even better in virtual camera pannings which are very common in PTZ, spherical and cillindrical surveillance cameras.

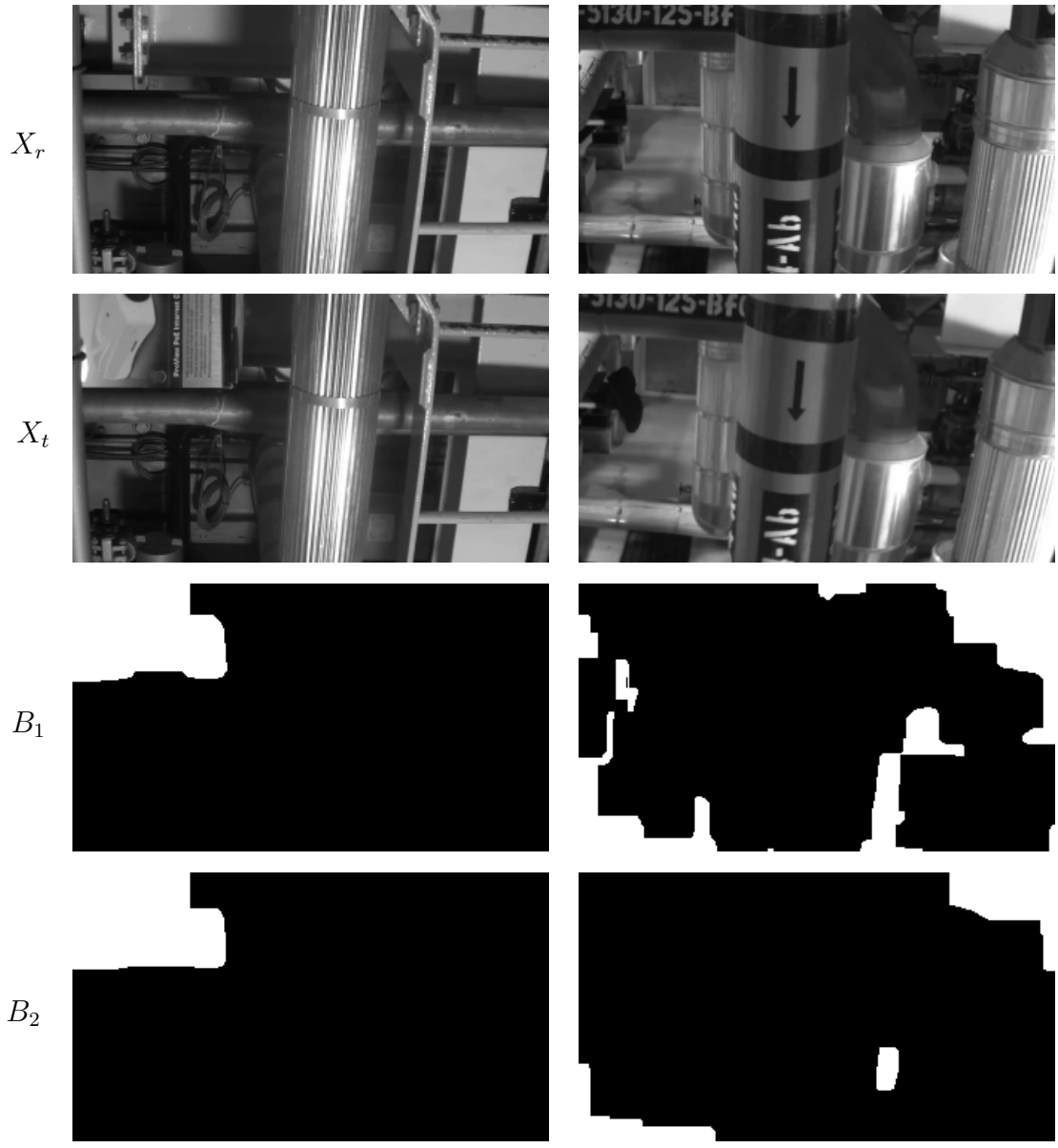


Figure 4.8: Result samples with different alignment conditions and different morphological parameters. B_1 row, processed with $\mu_1 = 4$ and $\mu_2 = 32$ and B_2 row processed with $\mu_1 = 12$ and $\mu_2 = 45$. Other parameters were set $\gamma = 6.938$, $\beta = 0.125$, $\omega = 12$, $\kappa = 1$.

Chapter 5

Domain-Transformable Sparse Representation

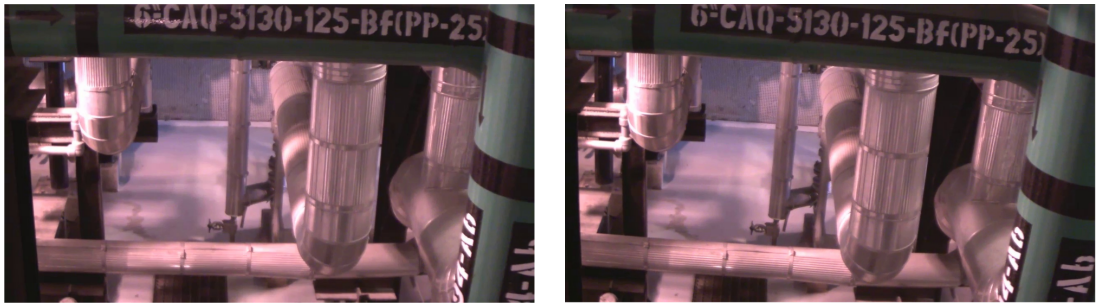
In this chapter we propose a novel sparse-representation algorithm for anomaly detection with moving-cameras. Section 5.1 discuss the use of domain-tranformations to cope with sample misalignments due to possible camera trepidations, while Section 5.2 enlights the details on how other PCA derived techniques employ domain-tranformations. Section 5.3 describes how to adapt domain-transformations to our solution and Section 5.4 compares our proposed method against several state-of-the-art method and discuss the results.

5.1 Domain-Transformations to correct sample misalignments

The algorithm presented in Chapter 4 tries to model the moving background as a blockwise low-rank factorization of the reference video matrix, plus a nonlinear sparse residue. For small perturbations in the camera path, this technique can successfully extract the target video foreground, exploiting the strong correlation between consecutive frames in both the reference and target videos and also between their corresponding frames, that is, frames in the reference and target videos that cover the same spot. Such a method has the great advantage of obviating the necessity of geometric registration of each frame. This is extremely useful when the perturbations of the camera movement are not so large, or even in synthetic moving video like "virtual pannings" generated from PTZ cameras or alike, saving a considerable computational time.

However, in many moving-camera practical scenarios, it is quite difficult to avoid eventual geometric misalignments between corresponding frames, as can be seen in figure 5.1. Cameras attached to moving platforms may suffer from this problem

since these devices may have to deal, besides normal camera vibration, with path irregularities or unpredicted weather conditions, for example.



(a) reference (b) target

Figure 5.1: Example of significant pose misalignment due to camera rotation in corresponding frames of reference and target videos in the VDAO dataset. Notice the different angles between the large pipe at the top with the horizontal axis.

As pointed in Section 3.3, some video sequences in the VDAO database contain severe misalignments between the target and reference samples. This issue ocurred during the acquisition phase of the database due to an excessive mechanical friction between the moving robot and the rail track, yielding to considerable shaking and rotation of camera attached to robot.

In this chapter we discuss the use of geometric domain-transformations to model the effect over the samples caused by the camera movement and rotation referred above. In short, the idea is to include a transformation term in the sparse representation procedure described in Eq. (4.2), updating the optimization algorithm to reflect these changes. The use of domain transformations together with l_1-optimization was done successfully in many recent works [28, 44–47]. The adaptation proposed in this work is inspired by these strategies.

5.2 A closer look at Robust Alignment

It is well known in literature that pixelwise alignment is a critical assumption when dealing with sparse representation and subspace modeling techniques. The presence of misalignments in the samples, either in the training or testing sets, may compromise the method's efficiency. Depending of the magnitude of the misalignment, it might break some linearity assumption over the samples or simply yield enough error to make the optimization unable to find a sparse solution to the problem. As said before, some recent work on l_1-optimization have proposed effective solutions to deal with sample misalignment. Although these new methods basically share the same approach, their minor differences are dependent on each problem particularities. The robust alignment approach will be reviewed here and adapted to the problem at hand on the next section.

5.2.1 Deformable SRC

Recently, Wright et. al. [26, 27] presented a robust face-recognition framework based on sparse representation-based classification (SRC), that achieved a remarkable performance compared to the best techniques at the time. The success of the method relies on harnessing the expected sparsity of a coefficient vector when trying to represent a testing sample as a linear combination of the columns of a given dictionary matrix. Let this matrix be $A = [A_1, A_2, \dots A_k]$, where each submatrix A_i is a list of training samples of a given subject i, which are present in the training dataset. These samples are basically pre-aligned face images, stacked into the columns of A. In this notation, given a test image y, the representation is done by finding the sparsest coefficient vector x such that $y = Ax$. The sparsity of x is assumed since the tested individual in y, say subject i, should only have non-zero coefficients at the part of x corresponding to A_i in A. To cope with possible occlusions appearing in y (e.g. sunglasses, hats, beards, closed eyes, etc), which are assumed to be sparse in terms of the image basis, an extra component e is added to the equation, that will be used further as an optimization constraint. Assuming that y was previously registered, the optimization algorithm searches for the sparsest vectors x and e, such that $y = Ax + e$, by minimizing the l_1-norm of both x and e. This can be summarized by

$$\min_{x,e} \|x\|_1 + \|e\|_1, \quad \text{s.t.} \quad y = Ax + e. \tag{5.1}$$

However, in pratical recognition systems, the tested face in y might not be perfectly aligned to the ones in the training set. An automatic SRC solution should also consider the registration of the testing sample in order to work efficiently. For example, an application that searches for an individual's face in photos of a social network or a security system that grants access using biometric recognition may have to deal with uncontrolled and unregistered input images.

A solution for that issue is proposed by Wagner et. al. [28], by considering that the testing samples are actually warped observations of the subject's face with respect to its pose in the training set. In this sense, let y^* be a well-aligned test image, which means that y^* is in the range of some A_i. Now let $\mathbb{T}$ be some transformation group that supposedly acts in the image domain. Assuming that y is a warped observation of y^*, there must exist a transformation $\tau^* \in \mathbb{T}$, such that $y^* = y \circ \tau^*$. More precisely, assuming the presence of occlusions in y, then there exists an unwarped error vector e^* in the same domain as y^*, such that $y^* + e^* = y \circ \tau^*$.

In practice, $\mathbb{T}$ is chosen to be a parametric, finite-dimensional, transformation group. Regularly known choices are the group of *translations*, *similarities*, the *affine transformations* and *planar homographies*. For implementation purposes, $\mathbb{T}$ can also

be identified as the vector space $\mathbb{R}^p$, where p is the number of needed parameters to describe any $\tau \in \mathbb{T}$.

Considering the possible warp in y^* and using this same notation, this leads to a direct extension of problem (5.1) as:

$$\min_{x,e,\tau} \|x\|_1 + \|e\|_1 \quad \text{s.t.} \quad y \circ \tau = Ax + e, \tag{5.2}$$

where τ^* is also searched in $\mathbb{T}$ for the sparsest representation of y^*. Notice that the modified problem is very similar to what was done in (2.13) in the RASL method (back in Section 2.2.2). Unfortunatelly, the optimization proposed in (5.2) is non-convex and hard to solve due the non-linearity of τ. However, for small variations of τ, it is possible to approximate this constraint in (5.2) by linearizing $y \circ \tau$ with the current estimate of τ such that

$$y \circ \tau + J\Delta\tau = Ax + e, \quad J = \frac{\partial}{\partial \tau}\left(y \circ \tau\right), \tag{5.3}$$

where J is the Jacobian of the image vector y with respect to τ. Instead of solving for τ directly, a small variation $\Delta\tau$ about the current estimate is searched in place, and problem (5.2) can be reformulated as

$$\min_{x,e,\Delta\tau} \|x\|_1 + \|e\|_1 \quad \text{s.t.} \quad y \circ \tau + J\Delta\tau = Ax + e. \tag{5.4}$$

The solution searched in problem (5.2) can be achieved by repeatedly solving (5.4) and then updating τ and J at each interation step. This representation should work fine if the samples are well aligned between the different A submatrices. This means that the samples in A_i must be well aligned with the samples in A_j for all $i, j \in \{1, 2, \ldots, k\}$.

In practice, the whole optimization can be computed within two loops. The outer loop updates the Jacobian J with the current estimate of τ, and updates τ by a $\Delta\tau$. This small variation step is recurrently computed inside the inner loop, which estimates $\Delta\tau$ along with the sparse coefficient x and the sparse error e, by solving (5.4). An initial transformation τ_0 must be provided as an initial condition for the algorithm. In the face recognition problem, a good starting point is to use a generic face detector to give an approximate position of the face sought in the input image. In practice, the test vector $y \circ \tau$ is a transformed window inside the original input image y. An additional procedure proposed by the deformable SRC authors is to normalize the training images at each iteration of the outer loop to avoid degenerate solutions like zooming to a dark region of the image. The steps described in Algorithm 3 summarize the procedure to solve problem (5.4) iteratively for a global representation among the subjects. The output x obtained

Algorithm 3 - SRC with transformed domain (global representation) [28]

input: Training images $A = [A_1, A_2, \dots A_k] \in \mathbb{R}^{n \times m}$, test image $y \in \mathbb{R}^m$, transformation group $\mathbb{T}$ and initial transformation $\tau_0 \in \mathbb{T}$

$\tau \leftarrow \tau_0$

while not converged **do**

step 1: normalize the image vector y and compute the Jacobian matrix J

$$\hat{y}(\tau) \leftarrow \frac{y \circ \tau}{\|y \circ \tau\|_2}, \quad J \leftarrow \frac{\partial}{\partial \eta} \hat{y}(\eta)\Big|_{\eta=\tau};$$

step 2 (inner loop): solve the linearized convex optimization (5.4):

$$(x^*, e^*, \Delta\tau^*) \leftarrow \arg\min_{x,e,\Delta\tau} \|x\|_1 + \|e\|_1 \quad \text{s.t.} \quad y \circ \tau + J\Delta\tau = Ax + e;$$

step 3: update transformation τ

$$\tau \leftarrow \tau + \Delta\tau^*;$$

step 4: compute convergence conditions

end while

ouput: solution x^*, e^* and τ^* to problem (5.2).

in this procedure is used to perform recognition in the same way proposed in [26, 27], observing which subject concentrates more coefficients to represent the testing face. The work in [28] also describes how to deal when faces between the submatrices are not well-aligned, but it deviates from the scope of this work. We have presented here a simplified version of the algorithm to illustrate how domain transformations can be incorporated into the l_1-optimization framework.

5.2.2 RASL

The deformable SRC case is a good starting point to understand how to adapt domain transformations with convex programming in order other kinds of problems. The RASL technique can be used to perform batch alignment on a set of correlated images D in the way it that it was seen in Section 2.2.2. Using the same approach as in last subsection, the relaxed version of problem described in Equation (2.13) is given by

$$\min_{A,E,\tau} \|A\|_* + \lambda\|E\|_1 \quad \text{s.t.} \quad D \circ \tau = A + E, \tag{5.5}$$

where $\|A\|_*$ is the nuclear norm of A, defined by $\text{tr}(\sqrt{A^*A})$, that works as the convex surrogate for $\text{rank}(A)$, and $\tau = [\tau_1|\dots|\tau_n] \in \mathbb{R}^n$ represent all the τ_i transformations

Algorithm 4 - RASL (outer loop) [46]

input: Vectorized images $I_1, \ldots, I_n \in \mathbb{R}^m$, initial transformations $\tau_1 \ldots \tau_n \in \mathbb{T}$ for each respective image, where $\mathbb{T}$ is the finite-dimensional transformation, and weight $\lambda > 0$

while not converged **do**

(step 1) compute Jacobian matrices for each τ_i

$$J_i \leftarrow \frac{\partial}{\partial \zeta}\left(\frac{I_i \circ \zeta}{\|I_i \circ \zeta\|_2}\right)\Bigg|_{\zeta=\tau_i}, \quad \text{for } i = 1, \ldots, n;$$

(step 2) warp and normalize the images in D matrix

$$D \circ \tau \leftarrow \left[\frac{I_1 \circ \tau_1}{\|I_1 \circ \tau_1\|_2} | \ldots | \frac{I_n \circ \tau_n}{\|I_n \circ \tau_n\|_2}\right];$$

(step 3) solve linearized convex optimization (inner loop):

$$(A^*, E^*, \Delta\tau^*) \leftarrow \arg\min_{A,E,\Delta\tau} \|A\|_* + \lambda\|E\|_1 \quad \text{s.t.}$$

$$D \circ \tau + \sum_{i=1}^{n} J_i \Delta\tau \epsilon_i \epsilon_i^\top = A + E.$$

(step 4) update transformations $\tau \leftarrow \tau + \Delta\tau$

end

ouput: solution A^*, E^* and τ^* to problem 5.5.

of a p-parameter group acting on each one of the n frames. Now, assuming that $\Delta\tau = [\Delta\tau_1 | \ldots | \Delta\tau_n] \in \mathbb{R}^n$ is a small variation on τ, one can linearly approximate the constraint in Equation (5.5) about the current estimate of τ such that

$$D \circ (\tau + \Delta\tau) \approx D \circ \tau + \sum_{i=1}^{n} J_i \Delta\tau \epsilon_i \epsilon_i^\top = A + E, \tag{5.6}$$

where the ϵ_i represent the canonical basis for $\mathbb{R}^n$, which leads to

$$\min_{A,E,\Delta\tau} \|A\|_* + \lambda\|E\|_1 \quad \text{s.t.} \quad D \circ \tau + \sum_{i=1}^{n} J_i \Delta\tau \epsilon_i \epsilon_i^\top = A + E. \tag{5.7}$$

Using this formulation, it its now possible to define the outer loop of RASL, in the same manner as in last subsection. This procedure can be seen in Algorithm 4. Notice that the main cost of the outer loop is obviouly in the optimization performed inside the inner loop. Hopefully, the optimization in step 3 can be efficiently solved using the Augmented Lagrange Multiplier (ALM) method. By making use of the

Algorithm 5 - RASL (inner loop) [46]

input: $A^0 \in \mathbb{R}^{m\times n}$, $E^0 \in \mathbb{R}^{m\times n}$, $\Delta\tau^0 \in \mathbb{R}^{p\times n}$, $\lambda > 0$

while not converged **do**
$(U, \Sigma, V) = \text{svd}\left[D \circ \tau + \sum_{i=1}^{n} J_i \Delta\tau_k \epsilon_i \epsilon_i^\top + \frac{1}{\mu_k} Y_k - E_k\right]$;
$A_{k+1} = U\mathcal{S}_{\frac{1}{\mu_k}}[\Sigma]V^\top$;
$E_{k+1} = \mathcal{S}_{\frac{\lambda}{\mu_k}}\left[D \circ \tau + \sum_{i=1}^{n} J_i \Delta\tau_k \epsilon_i \epsilon_i^\top + \frac{1}{\mu_k} Y_k - A_{k+1}\right]$;
$\Delta\tau_{k+1} = \sum_{i=1}^{n} J_i^\dagger \left(A_{k+1} + E_{k+1} - D \circ \tau - \frac{1}{\mu_k} Y_k\right) \epsilon_i \epsilon_i^\top$;
$Y_{k+1} = Y_k + \mu_k h(A_{k+1}, E_{k+1}, \Delta\tau_{k+1})$;
$\mu_{k+1} = \rho\mu_k$;
end while

ouput: solution A^*, E^* and $\Delta\tau^*$ to problem 5.7.

following auxiliary function

$$h(A, E, \Delta\tau) = D \circ \tau + \sum_{i=1}^{n} J_i \Delta\tau \epsilon_i \epsilon_i^\top - A - E,$$

we can define the augmented Lagrangian function for problem 5.7 as

$$\mathcal{L}_\mu(A, E, \Delta\tau, Y) = \|A\|_* + \lambda\|E\|_1 + \langle Y, h(A, E, \Delta\tau)\rangle + \frac{\mu}{2}\|h(A, E, \Delta\tau)\|_F^2, \quad (5.8)$$

where $Y \in \mathbb{R}^{m\times n}$ is a matrix representing the Lagrange multipliers and μ a penalty factor in the optimization. The ALM problem can be solved by minimizing the Lagrangian function and updating the Lagrange multipliers considering the penalty factor as follows

$$\begin{aligned} (A_{k+1}, E_{k+1}, \Delta\tau_{k+1}) &= \arg\min_{A,E,\Delta\tau} \mathcal{L}_{\mu_k}(A, E, \Delta\tau, Y_k) \\ Y_{k+1} &= Y_k + \mu_k h(A_{k+1}, E_{k+1}, \Delta\tau_{k+1}). \end{aligned} \quad (5.9)$$

Since it is infeasible to solve this problem directly, it is used an alternating strategy to approximate the result by minimizing one unknown at a time:

$$\begin{aligned} A_{k+1} &= \arg\min_A \ \mathcal{L}_{\mu_k}(A, E_k, \Delta\tau_k, Y_k), \\ E_{k+1} &= \arg\min_E \ \mathcal{L}_{\mu_k}(A_{k+1}, E, \Delta\tau_{k+1}, Y_k), \\ \Delta\tau_{k+1} &= \arg\min_{\Delta\tau} \ \mathcal{L}_{\mu_k}(A_{k+1}, E_{k+1}, \Delta\tau, Y_k). \end{aligned} \quad (5.10)$$

Fortunately, each of these steps have a closed-form solution can be solved efficiently. This procedure is summarized in Algorithm 5. Each $J_i^\dagger$ represents the pseudoinverse of J_i.

5.3 Adapting Domain-Transformations for Sparse-Representation of Moving Camera Sequences

Let $X_r \in \mathbb{R}^{m \times n_r}$ and $X_t \in \mathbb{R}^{m \times n_t}$ be the reference and target matrices, respectively, containing information about their corresponding video sequences. Let m be the number of pixels in the video frames, and n_r and n_t the number of frames in the reference and target videos, respectively. In a first moment, let us assume an ideal scenario where both reference and target sequences were acquired with exactly the same camera path and pose with similar internal and external setups. Under these conditions, it is fair to assume that every column in X_t has at least one corresponding column in X_r, leading to the problem given by

$$\min_{W,E} \|W\|_1 + \lambda \|E\|_1, \quad \text{s.t.} \quad X_t = X_r W + E, \tag{5.11}$$

where $W \in \mathbb{R}^{n_r \times n_t}$ is a coefficient matrix, which describes the relations of the columns of X_t and X_r, and $E \in \mathbb{R}^{m \times n_t}$ is the error term, which has the same dimensions of X_t. In this problem, the factor λ is used to balance the importance of the two minimized terms and may be adjusted considering the expected amount of sparsity in W and E. Notice that Eq. (5.11) is basically Eq. (4.2), but under a different context where the representation is direct, and not made on top of a subspace model.

Now, let us make an additional modification by breaking the assumption that X_t and X_r are perfectly aligned, since there are uncontrolled camera shaking and rotation. Since images acquired at the same center of projection can be related by a homography transformation [39], it is possible to consider that the target observations in X_t are in a different geometric domain with respect to X_r. This assumption allows one to model the camera shaking as a geometric transformation applied to a domain where corresponding target and reference samples are aligned. In this sense, Eq. (5.11) becomes

$$\min_{W,E,\tau} \|W\|_1 + \lambda \|E\|_1, \quad \text{s.t.} \quad X_t \circ \tau = X_r W + E, \tag{5.12}$$

where $\tau = [\tau_1 \dots \tau_{n_t}]$ is an vector of domain-transformations, in the same fashion as the discussed in Section 2.2. Each entry of τ acts on its corresponding column of X_t, that represents the observed target samples. For implementation purposes, τ is represented by a $p \times n_t$ matrix where each column is a vector with the p parameters necessary to describe each transformation acting on X_t.

Although, at a first glance one might think this development is a new imple-

mentation of [47], since it appears that the domain transformations are used in the same way, a thorough inspection of the proposed formulation shows that both X_r and X_t are fixed in the present formulation. Also, it would come naturally from [47] that the X_r matrix would be modified by the transforms, since it plays a similar role in the present work as the model, as well as the A matrix in RASL. We chose, however, to apply the transformations over X_t, since we assume here that X_r is well known by the system and is considered to be the best representation available to the background model. As for the X_t matrix we assume it might suffer from misalignments that should be corrected before the decomposition is performed.

In this problem, the composition of a geometric domain-transformation with X_t breaks the linearity of the the optimization constraint that appears on the right side of Eq. (5.12). However, for a small variation $\Delta\tau = [\Delta\tau_1|\dots|\Delta\tau_{n_t}] \in \mathbb{R}^{p\times n_t}$ of τ, it is possible to approximate this constraint by linearizing $X_t \circ \tau$ with the current estimate of τ, in a similar way it is done in [28] and [45], such that

$$X_t \circ (\tau + \Delta\tau) \approx X_t \circ \tau + \sum_{i=1}^{n_t} J_i \Delta\tau \epsilon_i \epsilon_i^\top = X_r W + E, \tag{5.13}$$

where the ϵ_i represent the canonical basis for $\mathbb{R}^{n_t}$ and

$$J_i = \frac{\partial}{\partial\zeta}\left(\frac{(X_t)_i \circ \zeta}{\|(X_t)_i \circ \zeta\|_2}\right)\Bigg|_{\zeta=\tau_i} \in \mathbb{R}^{m\times p} \tag{5.14}$$

is the Jacobian of the i-th column of the target matrix X_t with respect to τ_i. With these definitions, we can write the following the modified optimization problem

$$\begin{aligned} &\min_{W,E,\Delta\tau} \|W\|_1 + \lambda\|E\|_1, \\ &\text{s.t.} \quad X_t \circ \tau + \sum_{i=1}^{n_t} J_i \Delta\tau \epsilon_i \epsilon_i^\top = X_r W + E. \end{aligned} \tag{5.15}$$

Hence, the solution of Eq. (5.12) can be achieved by repeatedly solving (5.15) and then updating τ at each iteration step. In practice, the whole optimization of the so-called moving-camera domain-transformation sparse representation (mcDTSR) algorithm can be computed within two loops. In the outer loop, the Jacobian matrices J_i are computed based on the current estimate of τ. Then, the columns of $X_t \circ \tau$ are normalized to avoid undesired trivial solutions, like, for example, zooming to a black pixel of a given frame of X_t, that will end up with a null column of W. Only after that the inner loop is performed by solving (5.15). At last, τ is updated by the $\Delta\tau$ increment that is recurrently computed in the inner loop, which also estimates the sparse coefficient W and the error E. We have empirically observed that the relative change of the objective function is a good stopping criterion for the

Algorithm 6 - Domain-transformable sparse representation for moving camera videos (mcDTSR): outer loop

input: Reference matrix $X_r \in \mathbb{R}^{m \times n_r}$, target matrix $X_t \in \mathbb{R}^{m \times n_t}$, inital transformation vector $\tau = [\tau_1 \dots \tau_{n_t}] \in \mathbb{R}^{p \times n_t}$ and weight $\lambda > 0$

while not converged (Eq. (5.16) is not satisfied) **do**

(step 1) compute Jacobian matrices for each τ_i

$$J_i \leftarrow \frac{\partial}{\partial \zeta} \left(\frac{(X_t)_i \circ \zeta}{\|(X_t)_i \circ \zeta\|_2} \right) \Bigg|_{\zeta = \tau_i}, \quad \text{for } i = 1, \dots, n_t;$$

(step 2) warp and normalize the images in X_t matrix

$$X_t \circ \tau \leftarrow \left[\frac{(X_t)_1 \circ \tau_1}{\|(X_t)_1 \circ \tau_1\|_2} \ \cdots \ \frac{(X_t)_{n_t} \circ \tau_{n_t}}{\|(X_t)_{n_t} \circ \tau_{n_t}\|_2} \right];$$

(step 3) solve the linearized convex optimization problem (inner loop):

$$(W^*, E^*, \Delta\tau^*) \leftarrow \arg \min_{W,E,\Delta\tau} \|W\|_1 + \lambda \|E\|_1 \quad \text{s.t.}$$

$$X_t \circ \tau + \sum_{i=1}^{n_t} J_i \Delta\tau \epsilon_i \epsilon_i^\top = X_r W + E.$$

(step 4) update the transformation vector:

$$\tau \leftarrow \tau + \Delta\tau$$

end

ouput: solution W^*, E^* and τ^* to problem (5.12).

outer loop, meaning that given a positive value ε_r, the outer loop is exited when

$$\frac{|\text{obj}_k - \text{obj}_{k-1}|}{|\text{obj}_k|} < \varepsilon_r, \quad \text{obj}_k = \|W_k\|_1 + \lambda \|E_k\|_1, \tag{5.16}$$

where k is the iteration index of the outer loop. In this procedure, an initial set of transformations τ_0 must be provided for the outer loop, along with X_r and X_t matrices. Since both matrices are supposed to be acquired under similar conditions, τ_0 is initially chosen to be a set of identity transforms. The proposed method for the mcDTSR outer loop is summarized in Algorithm 6. To efficiently solve Eq. (5.15) inside the inner loop, we shall make use of the augmented Lagrangian method [25]. By defining the following auxiliary function

$$h(W, E, \Delta\tau) = X_r W + E - X_t \circ \tau - \sum_{i=1}^{n} J_i \Delta\tau \epsilon_i \epsilon_i^\top, \tag{5.17}$$

it is possible to write the augmented Lagrangian function as

$$\begin{aligned}\mathcal{L}_\mu(W, E, \Delta\tau, Y) = \|W\|_1 + \lambda\|E\|_1 + \langle Y, h(W, E, \Delta\tau)\rangle \\ + \frac{\mu}{2}\|h(W, E, \Delta\tau)\|_F^2,\end{aligned} \tag{5.18}$$

where Y is Lagrange multiplier matrix and μ is a positive scalar. This can be solved by estimating both Y and the optimal solution iteratively [30] as follows

$$\begin{aligned}(W_{k+1}, E_{k+1}, \Delta\tau_{k+1}) &= \arg\min_{W,E,\Delta\tau} \mathcal{L}_{\mu_k}(W, E, \Delta\tau, Y_k), \\ Y_{k+1} &= Y_k + \mu_k h(W_{k+1}, E_{k+1}, \Delta\tau_{k+1}), \\ \mu_{k+1} &= \rho\mu_k,\end{aligned} \tag{5.19}$$

where μ_0 and ρ are tunable parameters and will be discussed later. To facilitate the solution of Eq. (5.19), we can break it into three new equations and approximate the result by minimizing one unknown at a time, such that

$$\begin{aligned}W_{k+1} &= \arg\min_{W} \ \mathcal{L}_{\mu_k}(W, E_k, \Delta\tau_k, Y_k), \\ E_{k+1} &= \arg\min_{E} \ \mathcal{L}_{\mu_k}(W_{k+1}, E, \Delta\tau_k, Y_k), \\ \Delta\tau_{k+1} &= \arg\min_{\Delta\tau} \ \mathcal{L}_{\mu_k}(W_{k+1}, E_{k+1}, \Delta\tau, Y_k).\end{aligned} \tag{5.20}$$

The great advantage of alternating the unknowns in (5.20) is that each one has a direct form of computation.

As in [32], W and E can be estimated by the soft-thresholding operator, defined as

$$\mathcal{S}_\gamma[A] = \operatorname{sign}(A) \cdot \max\{|A| - \gamma, 0\}, \tag{5.21}$$

where the sign and max operations are applied entrywise on the matrix A. By expanding the expressions in Eq. (5.20) using the same rationale underlying the development in [33], and using h as in defined in Eq. (5.17), we would have:

$$\begin{aligned}W_{k+1} &= \mathcal{S}_{\frac{1}{\mu_k}}\left[W_k - X_r^\top\left(h(W_k, E_k, \Delta\tau_k) + \tfrac{1}{\mu_k}Y_k\right)\right], \\ E_{k+1} &= \mathcal{S}_{\frac{\lambda}{\mu_k}}\left[E_k - \left(h(W_{k+1}, E_k, \Delta\tau_k) + \tfrac{1}{\mu_k}Y_k\right)\right], \\ \Delta\tau_{k+1} &= \mathcal{S}_{\frac{1}{\mu_k}}\left[\Delta\tau_k - \mathcal{J}^*\left(h(W_{k+1}, E_{k+1}, \Delta\tau_k) + \tfrac{1}{\mu_k}Y_k\right)\right],\end{aligned} \tag{5.22}$$

where $\mathcal{J}^*(\theta)$ is the adjoint of the functional $\mathcal{J}(\theta) = \sum_{i=1}^n J_i\theta\epsilon_i\epsilon_i^\top$ which is applied over $\Delta\tau$ in $h(W, E, \Delta\tau)$. However, in our application the $\Delta\tau$ is not assumed to be sparse, therefore we chose not to apply the soft threshold operator $\mathcal{S}_\gamma[\cdot]$ in its

Algorithm 7 - Domain-transformed sparse representation for moving camera videos (mcDTSR): inner loop

input: $W_0 \in \mathbb{R}^{n_r \times n_t}$, $E_0 \in \mathbb{R}^{m \times n_t}$, Q, $\overline{\Delta\tau}_0 = 0 \in \mathbb{R}^{p \times n_t}$, $\mu_0 > 0$, $\rho > 0$, $\lambda > 0$

let $h(W, E, \overline{\Delta\tau}) = X_r W + E - X_t \circ \tau - \sum_{i=1}^{n} Q_i \overline{\Delta\tau} \epsilon_i \epsilon_i^\top$

while not converged (Eq. (5.24) is not satisfied) **do**

$W_{k+1} = \mathcal{S}_{\frac{1}{\mu_k}} \left[W_k - X_r^\top \left(h(W_k, E_k, \overline{\Delta\tau_k}) + \frac{1}{\mu_k} Y_k \right) \right]$;

$E_{k+1} = \mathcal{S}_{\frac{\lambda}{\mu_k}} \left[E_k - \left(h(W_{k+1}, E_k, \overline{\Delta\tau_k}) + \frac{1}{\mu_k} Y_k \right) \right]$;

$\overline{\Delta\tau}_{k+1} = \overline{\Delta\tau}_k + \sum_{j=1}^{n} Q_j^\top \left(h(W_{k+1}, E_{k+1}, \overline{\Delta\tau_k}) + \frac{1}{\mu_k} Y_k \right) \epsilon_j \epsilon_j^\top$;

$Y_{k+1} = Y_k + \mu_k \, h(W_{k+1}, E_{k+1}, \overline{\Delta\tau}_{k+1})$;

$\mu_{k+1} = \rho \mu_k$;

end while

$\forall_i : \Delta\tau_i = R_i^{-1} \overline{\Delta\tau}_i$

ouput: solution W^*, E^*, and $\Delta\tau^*$ to problem (5.15).

update equation, replacing the last line of Eq. 5.22 by

$$\Delta\tau_{k+1} = \Delta\tau_k - \mathcal{J}^* \Big(X_r W_{k+1} + E_{k+1} - X_t \circ \tau - \sum_{i=1}^{n} J_i \Delta\tau_k \epsilon_i \epsilon_i^\top + \frac{1}{\mu_k} Y_k \Big). \quad (5.23)$$

The functional $\mathcal{J}^*(\theta)$ can be approximated by $\sum_{i=1}^{n} J_i^\top \theta \epsilon_i \epsilon_i^\top$, in our application.

Since the space-size parameter p is relatively small when compared to the frame resolution dimension m, the Jacobian matrices J_i are likely to be ill-conditioned, which may lead to numerical instability in the inner loop. To work this around, one may perform a QR factorization of the Jacobians, that is, $J_i = Q_i R_i$, and use orthogonal factors Q_i inside the inner loop in place of the Jacobians J_i. In this manner, the inner loop will output $\overline{\Delta\tau_i} = R_i \Delta\tau_i$ instead of $\Delta\tau_i$ for each component of $\Delta\tau$, also the inner loop will only see the Q_i components of each J_i. Since the R_i are invertible, $\Delta\tau$ can be easily computed [47].

The mcDTSR inner loop described in Algorithm 7 solves separately for both W and E, using the linearized alternating direction method with adaptive penalty (LADMAP) approach [30], differently from the approach in [47], where the ALM is applied. By expanding the Lagrangian using LADMAP one is able to reach a faster convergence [30]. The use of LADMAP is the reason μ_k is updated by a positive ρ (Eq. (5.19)). The value of ρ has influence on the compromise between approximation accuracy and the algorithm's running time. For the stopping criterion of the inner loop, one may consider the ratio between the Frobenius norm of h (which can be thought of as the residual of the cost function in Eq. (5.15)), and the norm of $X_t \circ \tau$

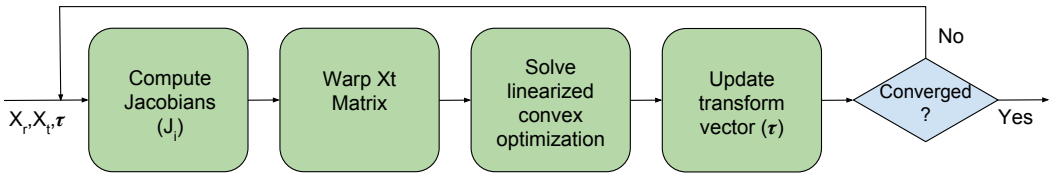


Figure 5.2: mcDTSR block diagram.

itself. More precisely, the inner loop will stop when

$$\frac{\|h(W_k, E_k, \Delta\tau_k)\|_F}{\|X_t \circ \tau\|_F} < \varepsilon_t, \tag{5.24}$$

where k is the current inner-loop iteration index.

Assuming that reference and target videos may have significant misalignments between its correspondent frames, working with the full video frames can make the warped frames of X_t present invalid pixels at the borders, that are the result of the mapping of pixels beyond the borders of X_t. As these invalid pixels can affect the algorithm convergence, a common practice is to work with a region-of-interest (ROI) window that is smaller than the full video frame, so that it can have some freedom to warp and avoid the mapping of pixels outside the frame's borders. Thus, X_t and X_r are in general ROI windows inside the full frames.

Figure 5.2 shows the block diagram for the proposed algorithm.

5.4 Experimental Results

In order to evaluate the detection efficiency of our proposed technique, two types of experiments were elaborated. The first is intended to qualitatively assess the application of the method on a problematic case, giving an introspective view of the method's components and how they evolve along the algorithm's iterations. The second one use assessing metrics to compare mcDTSR with the current state-of-art techniques. The VDAO-200 dataset described in Sec. 3.3.2 is used to perform the quantitative experiments in upcoming Section 5.4.3. Since the method promotes changes in the domain-space of the target video, some implementation details about how to measure the detection quality with the given ground truth annotations are also discussed in the next section.

5.4.1 Domain-transformation compensation

The VDAO database comes with ground truth annotations of the abandoned objects for every target-video frame, where the object positions are marked with rectangular bounding boxes. Since the abandoned objects have arbitrary shapes, working with bounding boxes can lead to results which are not very precise, that may mask the actual amounts of true and false positives.

This said, another relevant concern is that τ is computed with respect to the target video, so any evaluation metric should take into account the domain-transformation performed on each frame of X_t. Since a general transformation should change the annotated bounding boxes into quadrilateral polygons, we chose to maintain the target domain fixed and apply the inverse transformation τ^{-1} to the reference domain when carrying out the performance assessment. More precisely, applying τ^{-1} to both sides of the constraint in Eq. (5.12) leads to

$$X_t = X_r W \circ \tau^{-1} + E', \tag{5.25}$$

where $E' = E \circ \tau^{-1}$. To compute E' we consider the transformation applied to the whole image, and not only the region-of-interest. So, a general transformation on $X_r W$ may yield frames that contain zero values near their boundaries, since the image border may be overlapped by the resulting quadrilateral. To avoid dummy false positives, E' is set to zero in these problematic regions. The metrics described in the sequel are applied to this resulting error image, after all post-processing steps. These choices are motivated by simplicity and the possibility to compare our results with the other competing methods, since they act in the canonical geometric-domain of the target video.

5.4.2 Qualitative evaluation

In this section, we illustrate the advantage of including domain transformations into the optimization process. When the corresponding target and reference sequences have considerable levels of misalignment, the sparse representation of the target frames performs poorly, generally introducing several artifacts into the residual component E. If some algorithm that uses low-rank or sparse representation is used for detection purposes, this misalignment can yield a large number of false positive regions, possibly masking the actual presence of strange objects on the scene, compromising the practical applicability of such a method. In this sense, a simple experiment was designed to illustrate and qualitatively evaluate the gain in detection performance provided by the proposed algorithm. To this end, the main components of Eq. (5.15) will be inspected along the iterations of the mcDTSR outer loop de-

scribed in Algorithm 6, providing some insights about what is happening "under the hood".

For this task, we have selected an excerpt of the target video from the VDAO database entitled "Object 3 (shoe, position 3)". This sequence presents a case of significant misalignment with respect to its corresponding reference video, making any conventional method that is not tolerant to camera shaking not to perform well. A 50-frame snippet of this target video was selected together with a 100-frame snippet of the corresponding reference video, manually chosen such that the entire target excerpt can be represented by the reference one. It is important to point out that, although in this case the target-reference match is guaranteed, the algorithm has no information about which reference frames shall be used to represent an arbitrary target frame, nor about the parameters of camera tilt between these corresponding frames. To reduce the processing time, these video snippets where downsampled to a 320×180-pixel resolution and converted to grayscale, and the chosen regions-of-interest (ROI) were the 280×150-pixel central windows from each frame in both videos.

By considering only planar homographies to represent the domain transformation, one gets $p = 8$ by using a 4-point parametrization to describe the columns of τ. The parameter setup used in mcDTSR was $\lambda = 10^2$, $\rho = 1.01$, and $\mu_0 = 1.25/\|X_t \circ \tau\|$, following the values used in [34]. The inner loop tolerance for the stopping criterion was set to $\varepsilon_t = 10^{-4}$ and the outer loop tolerance ε_r was left loose. The idea was to observe how the magnitudes of the algorithm unknowns and metrics behave along a total of 55 outer-loop iterations.

At the post-processing detection stage, a simple thresholding procedure was performed by marking as foreground every entry of $|E|$ with intensity greater than $\beta = 0.125$, otherwise turning it as background. This value was chosen based on the adjusted values described in Table 4.1.

In Figs. 5.3(a), 5.3(b), and 5.3(c), it is possible to see the evolution of the l_1-norm of $\|W\|$, $\|E\|$, and $\|\Delta\tau\|$, respectively, across the outer-loop iterations. Independently of their magnitude ranges, one can clearly notice how the three norms evolve in time, converging to their final values after approximately 45 iterations.

However, the great strength of the proposed method can be noticed in Figs. 5.3(d), 5.3(e), and 5.3(f), where some detection metrics for the mcDTSR algorithm are displayed. All these three metrics were computed pixelwise, by comparing the binary mask video E', as given in Section 5.4.1, to the provided bounding-box ground truth from the VDAO database.

The behaviour shown by the true-positive rate (TPR) plot in Fig. 5.3(d) is explained by the fact that the ground truth bounding boxes are larger than the actual object. Thus, this plot represents the superposition of the actual false positives that

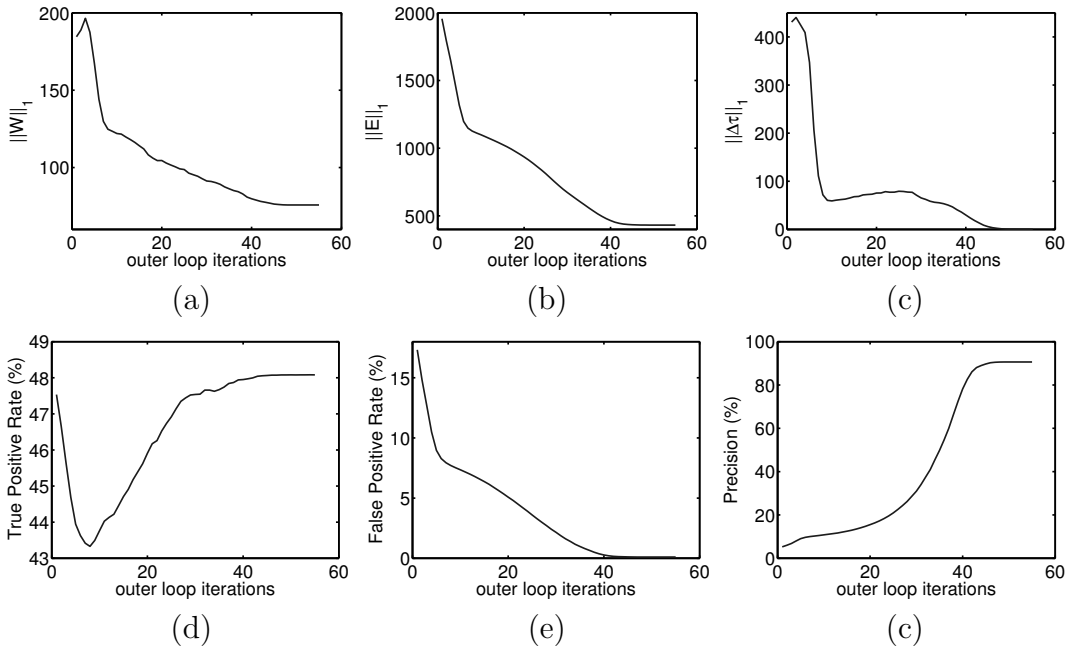


Figure 5.3: Evolution of several mcDTSR parameters and performance metrics along outer-loop iterations, illustrating improvement over time of proposed algorithm: (a) $\|W\|_1$; (b) $\|E\|_1$; (c) $\|\Delta\tau\|_1$; (d) True-positive detection rate; (e) False-positive detection rate; (f) Precision rate.

lie inside the bounding box being eliminated, promoting a decrease in the TPR, plus the actual object being increasedly detected. This can be appreciated by looking also at evolution of E over the outer-loop iterations in Fig. 5.4. This figure also explains the impressive false-positive rate (FPR) and precision plots depicted in Figs. 5.3(e) and Figs. 5.3(f), respectively, as a result from the improved geometric alignment between the target and reference frames. In fact, from the first outer-loop iteration ($l = 1$) to the last one ($l = 55$), more than 99% of the false positives were eliminated.

The geometric alignment can be crucial to the convergence of sparse representation methods. This is well illustrated by Figs. 5.5 and 5.6, which show the evolutions of the target ROIs and the estimated W, respectively. In fact, the improved geometric alignment provided by the transformation τ, as given in Fig. 5.5, enables a more robust and consequently more precise matrix factorization for the target video, as seen in Fig. 5.6.

5.4.3 Quantitative evaluation

For this experiment, are considered all 59 200-frame videos excerpts from the VDAO-200 subset, as given in Section 3.3. The parameter setup for the proposed mcDTSR algorithm are the same as in Section 5.4.2, with addition of the stopping criterion

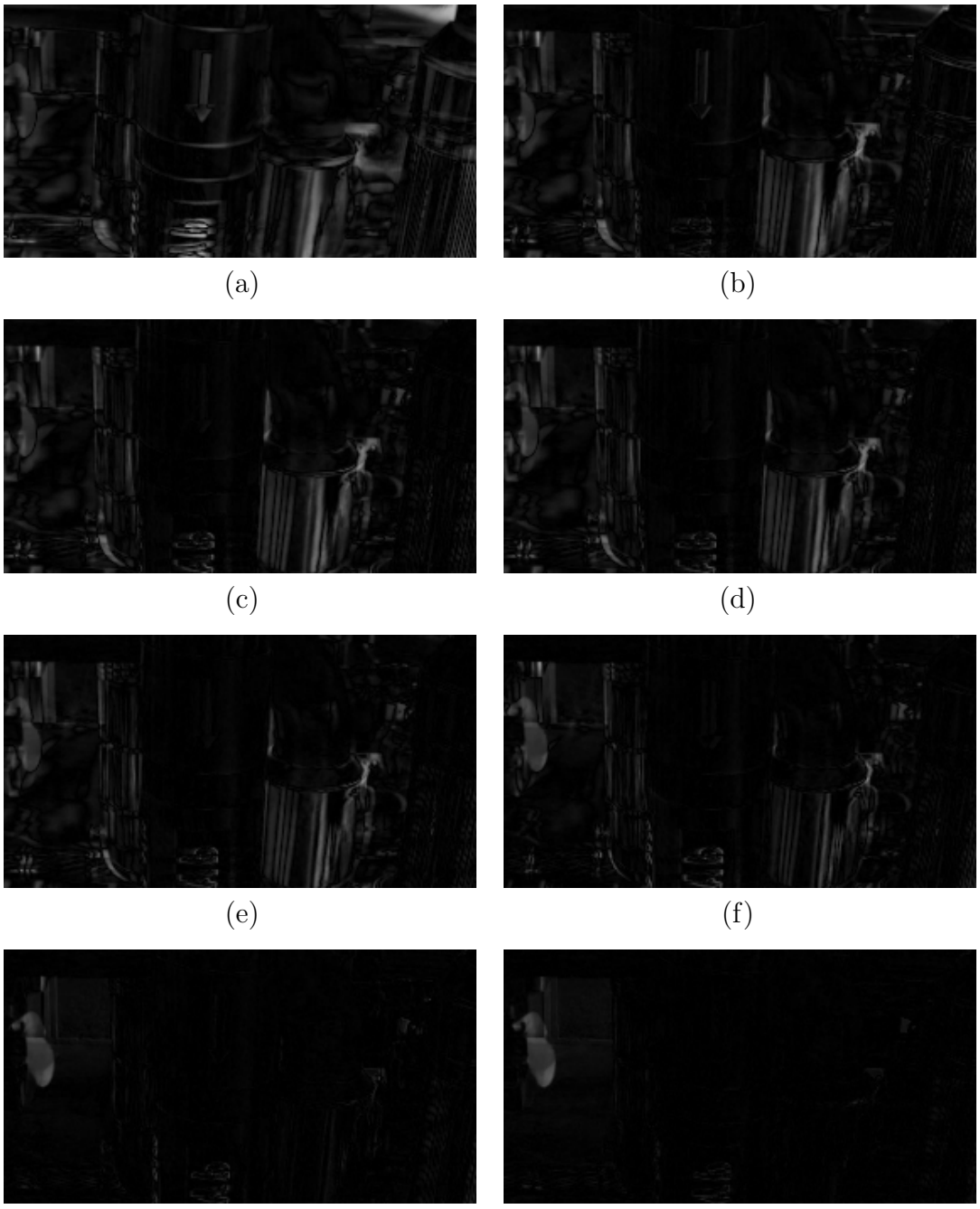


Figure 5.4: Evolution of residual matrix $|E|$ through selected mcDTSR outer-loop iterations l, for a fixed video frame: (a) $l = 1$; (b) $l = 5$; (c) $l = 10$; (d) $l = 15$; (e) $l = 20$; (f) $l = 30$; (g) $l = 40$; (h) $l = 55$. Notice that the gradual alignment between target and reference correspondences contributes for an impressive reduction of potential false-positives regions, and also for a more precise detection of the abandoned object.

set to $\varepsilon_r = 10^{-5}$. The post-processing detection stage is composed by a thresholding step on $|E'|$ with $\beta = 0.2$ (chosen empirically), followed by morphological open and then close operations with 2 and 4 pixel-wide, disk-shaped structuring elements, respectively. This is followed by a simple temporal voting using a 5 pixel-wide

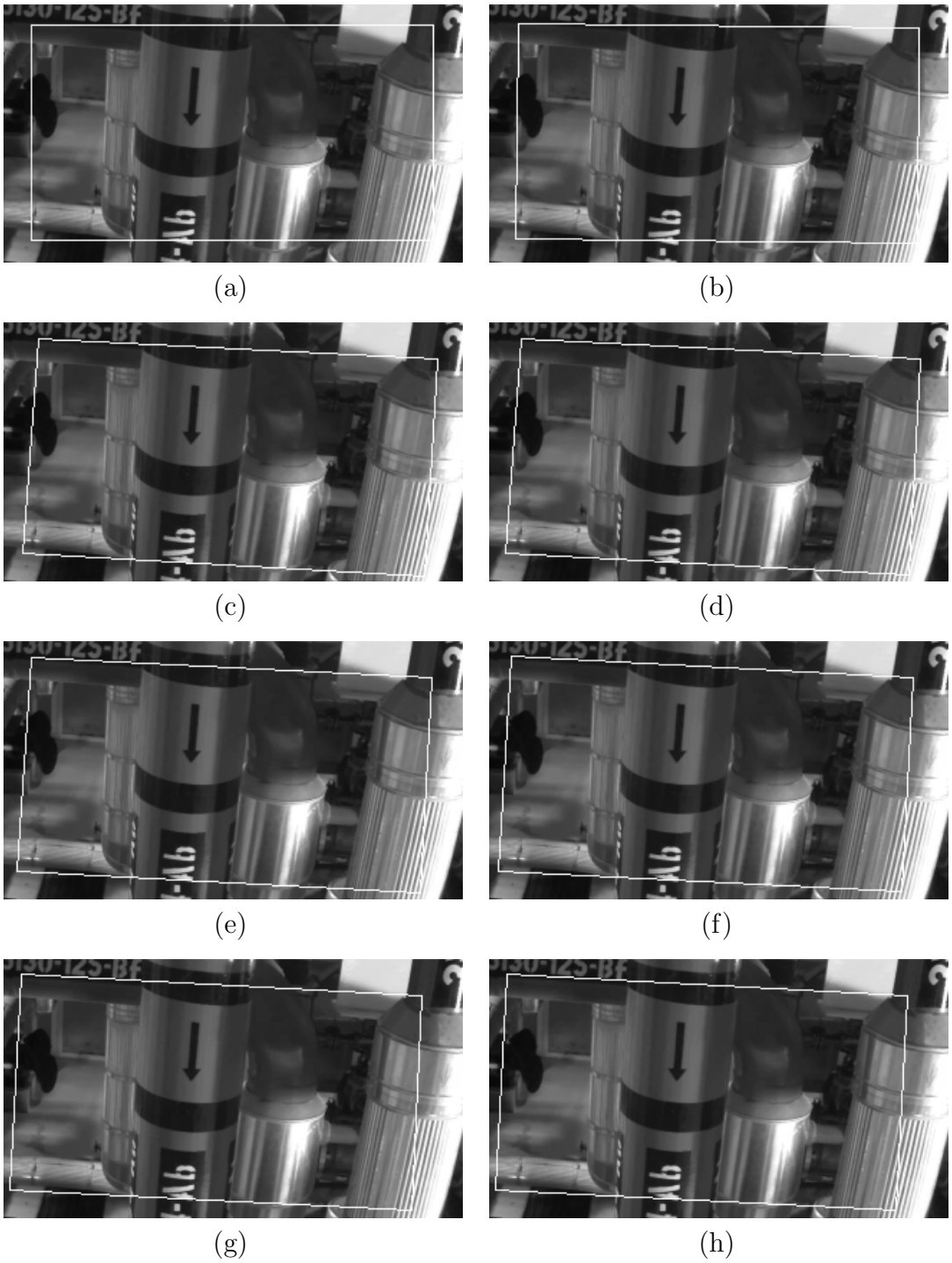


Figure 5.5: ROI evolution through selected mcDTSR outer-loop iterations l, for a fixed video frame: (a) $l = 0$ (initial ROI); (b) $l = 1$; (c) $l = 10$; (d) $l = 19$; (e) $l = 28$; (f) $l = 37$; (g) $l = 46$; (h) $l = 55$.

window, that turns the pixel on if more than half of the window is also on.

To assess the performance of the proposed mcDTSR method, the following metrics are employed: true positive rate (TP) and false positive rate (FP). A true positive happens when the detection blob has a non-empty intersection with the abandoned-object ground-truth bounding box, and a false positive happens when

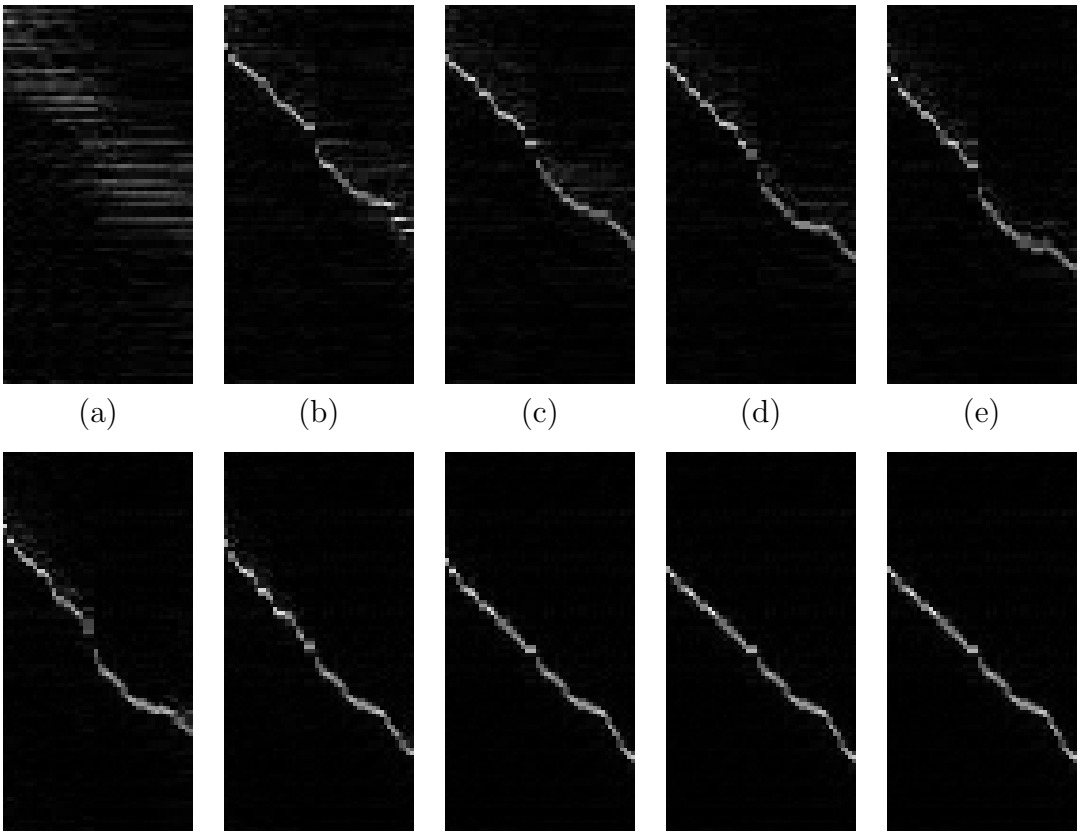


Figure 5.6: Evolution of weight matrix $|W|$ through selected mcDTSR outer-loop iterations l, for a fixed video frame: (a) $l = 1$; (b) $l = 7$; (c) $l = 13$; (d) $l = 19$; (e) $l = 25$; (f) $l = 31$; (g) $l = 37$; (h) $l = 43$; (i) $l = 47$; (j) $l = 55$. Notice how the target frames are incorrectly temporally correlated with the reference frames at the first iterations. The optimization gradually shifts the correlation to the correct frames, thanks to the transformations applied to the target video.

the detection blob and the ground-truth bounding boxes are disjoint. Another metric used for performance assessment is the DIS metric, that integrates both TP and FP, defined as

$$\mathrm{DIS} = \sqrt{(1 - \mathrm{TP})^2 + \mathrm{FP}^2}. \tag{5.26}$$

The DIS metric represents the minimum distance of the (TP,FP) point to the point of ideal behaviour (TP = 1 and FP = 0) on the TP×FP plane. The use of this metric allows direct comparison with the results in [31] and [36] for several state-of-the-art methods found in the literature, namely: the spatio-temporal composition for moving-camera detection (STC-mc) [37]; the detection of abandoned objects with a moving camera (DAOMC) [5]; the moving-camera background subtraction (MCBS) [38] the anomaly detection with a moving camera using multiscale video analysis (ADMULT) [31]; and the anomaly detection in moving-camera video sequences using principal subspace analysis (mcRoSuRe-A) [36]. The overall results for all these methods, including the proposed mcDTSR algorithm, are given in

Table 5.1.

Table 5.1: Comparison results of the proposed mcDTSR method with STC-mc, DAOMC, MCBS, and ADMULT, considering all the 59 single-object videos of the VDAO-200 database.

Video #	STC-mc [37]			DAOMC [5]			MCBS [38]			ADMULT [31]			mcRoSuRe-A [36]			mcDTSR		
	TP	FP	*DIS*	TP	FP	*DIS*	TP	FP	*DIS*	TP	FP	*DIS*	TP	FP	*DIS*	TP	FP	*DIS*
1	0.37	0.42	0.76	1.00	1.00	1.00	1.00	0.10	0.10	1.00	0.63	0.63	1.00	0.00	**0.00**	1.00	1.00	1.00
2	1.00	0.04	0.04	1.00	0.00	**0.00**	1.00	0.90	0.90	1.00	0.00	**0.00**	0.96	0.17	0.17	0.73	0.00	0.27
3	0.90	0.04	0.11	1.00	0.04	0.04	1.00	0.28	0.28	1.00	0.00	**0.00**	1.00	0.00	**0.00**	1.00	0.71	0.71
4	1.00	0.03	0.03	1.00	0.00	**0.00**	1.00	0.00	**0.00**	1.00	0.00	**0.00**	1.00	0.00	**0.00**	1.00	0.00	**0.00**
5	0.92	0.01	0.08	1.00	0.10	0.10	1.00	0.07	**0.07**	0.71	0.95	0.95	0.99	0.54	0.54	1.00	0.60	0.60
6	0.29	0.64	0.96	1.00	0.10	0.10	1.00	0.99	0.99	1.00	0.00	**0.00**	1.00	0.75	0.75	1.00	0.79	0.79
7	0.99	0.13	0.13	1.00	1.00	1.00	1.00	0.96	0.96	1.00	0.00	**0.00**	1.00	0.00	**0.00**	1.00	0.00	**0.00**
8	0.00	0.01	1.00	1.00	0.87	0.87	0.75	0.31	0.39	0.54	0.02	0.47	1.00	0.16	**0.16**	1.00	0.17	0.17
9	0.00	1.00	1.41	0.94	1.00	1.00	0.67	0.18	0.37	0.52	0.06	0.48	0.94	0.00	0.06	0.97	0.00	**0.03**
10	0.01	0.01	0.99	1.00	0.97	0.97	0.89	0.10	**0.15**	0.69	0.00	0.31	0.99	0.76	0.76	0.96	0.72	0.72
11	0.03	0.79	1.25	0.98	0.98	0.98	0.73	0.32	0.42	0.67	1.00	1.05	0.94	0.05	**0.07**	0.93	0.00	**0.07**
12	0.20	0.07	0.81	0.94	0.48	0.48	0.87	1.00	1.01	1.00	0.22	0.22	0.92	0.00	**0.08**	0.90	0.00	0.10
13	0.00	0.50	1.12	0.86	0.71	0.72	0.84	0.00	0.16	0.64	0.19	0.40	0.98	0.00	0.02	1.00	0.00	**0.00**
14	0.08	0.05	0.92	1.00	0.74	0.74	0.92	0.01	0.08	1.00	0.15	0.15	0.99	0.00	**0.01**	0.88	0.00	0.12
15	0.00	1.00	1.41	1.00	1.00	1.00	0.89	1.00	1.01	0.59	0.04	0.42	1.00	0.23	0.23	1.00	0.03	**0.03**
16	0.00	0.08	1.00	0.77	1.00	1.02	0.00	0.00	1.00	0.00	0.00	1.00	0.91	0.12	0.15	0.89	0.08	**0.13**
17	0.06	1.00	1.37	0.96	0.46	0.46	0.80	0.12	0.23	0.62	0.30	0.48	0.94	0.04	0.07	0.96	0.01	**0.04**
18	0.00	0.09	1.00	0.75	0.99	1.02	0.43	0.00	0.57	0.00	0.23	1.03	0.54	0.00	0.46	0.55	0.00	**0.45**
19	0.00	0.03	1.00	1.00	0.67	0.67	0.89	0.00	0.11	0.54	0.15	0.48	1.00	0.03	**0.03**	0.95	0.00	0.05
20	0.36	0.50	**0.81**	0.26	1.00	1.24	0.67	1.00	1.05	0.00	0.00	1.00	0.99	0.97	0.97	0.78	0.98	1.00
21	0.00	0.68	1.21	0.97	0.62	0.62	0.95	0.61	0.61	0.97	0.72	0.72	1.00	0.37	0.37	1.00	0.04	**0.04**
22	0.00	0.07	1.00	1.00	0.90	0.90	0.92	0.05	0.09	0.68	0.75	0.81	1.00	0.02	**0.02**	1.00	0.04	0.04
23	0.00	0.83	1.30	0.93	1.00	1.00	0.97	1.00	1.00	1.00	1.00	1.00	0.93	0.76	0.76	0.72	0.04	**0.28**
24	0.58	0.93	1.02	0.00	1.00	1.41	0.00	0.73	1.24	0.00	0.00	1.00	0.69	1.00	1.05	0.12	0.00	**0.88**
25	0.00	0.02	1.00	1.00	0.90	0.90	0.58	0.00	**0.43**	0.56	0.55	1.00	0.51	0.00	0.50	0.54	0.01	0.47
26	0.00	0.06	1.00	1.00	0.54	0.54	0.87	0.05	0.14	0.64	0.01	0.70	0.99	0.07	**0.07**	1.00	0.53	0.53
27	0.26	0.34	0.82	1.00	0.72	0.72	1.00	1.00	1.00	1.00	0.10	0.36	1.00	0.41	0.41	1.00	0.02	**0.02**
28	0.01	0.01	1.00	1.00	0.89	0.89	1.00	0.00	**0.00**	1.00	0.00	**0.00**	0.64	0.27	0.45	0.27	0.00	0.74
29	0.00	0.14	1.01	0.91	0.98	0.98	0.76	0.02	**0.24**	0.68	0.01	0.32	0.43	0.81	0.99	0.00	0.00	1.00
30	0.00	0.01	1.00	1.00	0.97	0.97	0.80	0.49	0.53	0.56	0.00	0.44	1.00	0.36	**0.36**	1.00	0.55	0.55
31	0.00	0.01	1.00	1.00	0.61	**0.61**	0.87	0.80	0.81	0.61	0.55	0.67	0.95	0.81	0.81	0.95	1.00	1.00
32	0.00	0.01	1.00	1.00	0.78	0.78	0.83	0.00	0.17	0.32	0.00	0.68	1.00	0.01	**0.01**	0.99	0.02	0.02
33	0.78	0.81	**0.83**	0.83	1.00	1.01	1.00	1.00	1.00	1.00	1.00	1.00	0.96	1.00	1.00	0.92	1.00	1.00
34	0.00	0.02	1.00	1.00	0.69	0.69	0.70	0.00	0.30	0.56	0.00	0.44	0.95	0.02	0.05	0.97	0.03	**0.04**
35	0.00	0.97	1.39	0.97	0.62	0.62	0.87	0.82	0.83	0.62	0.01	0.38	0.94	0.81	0.81	0.96	0.00	**0.04**
36	0.24	1.00	1.26	0.02	1.00	1.40	1.00	1.00	1.00	1.00	1.00	1.00	0.98	1.00	1.00	0.95	0.07	**0.08**
37	0.43	0.18	0.59	0.99	1.00	0.96	0.93	0.00	0.07	0.93	0.00	0.07	0.99	0.00	**0.01**	0.97	0.00	0.03
38	0.00	1.00	1.41	1.00	0.99	0.99	0.71	0.05	0.30	0.44	0.00	0.56	0.96	0.00	**0.04**	0.72	0.00	0.28
39	0.09	0.04	0.91	0.91	1.00	1.00	0.84	0.93	0.94	1.00	0.25	**0.25**	0.91	1.00	1.00	0.92	1.00	1.00
40	0.56	0.44	0.92	1.00	0.95	0.95	1.00	0.56	0.56	1.00	0.14	**0.14**	0.92	0.28	0.29	1.00	1.00	1.00
41	0.00	0.78	1.27	0.64	0.99	1.05	0.88	0.87	0.87	0.87	1.00	1.01	0.96	1.00	1.00	1.00	0.03	**0.03**
42	0.00	1.00	1.41	0.96	0.96	0.96	0.88	0.91	0.91	0.49	0.00	0.51	0.96	0.00	0.04	0.99	0.00	**0.01**
43	0.00	0.08	1.00	0.72	1.00	1.04	0.14	0.00	0.86	0.00	0.00	1.00	0.93	0.15	0.16	0.93	0.11	**0.13**
44	0.00	0.19	1.02	0.96	1.00	1.00	0.73	0.14	0.31	0.63	0.00	0.37	0.92	0.43	0.43	0.95	0.04	**0.06**
45	0.15	0.92	1.25	0.01	1.00	1.41	0.82	1.00	1.02	1.00	1.00	1.00	0.71	1.00	1.04	0.37	0.41	**0.75**
46	0.00	0.43	1.09	0.93	0.97	0.97	0.95	0.79	0.79	0.99	0.14	0.14	0.92	0.01	**0.08**	0.91	0.00	0.09
47	0.01	0.20	1.01	1.00	1.00	1.00	0.93	0.00	**0.07**	0.91	0.22	0.24	0.98	0.30	0.30	0.97	0.26	0.26
48	0.00	0.01	1.00	0.96	0.97	0.97	0.72	0.16	0.32	0.42	0.00	0.58	0.96	0.03	0.05	0.98	0.00	**0.02**
49	0.00	0.04	1.00	1.00	0.99	0.99	1.00	0.06	**0.06**	0.93	0.00	0.07	1.00	0.24	0.24	1.00	0.76	0.76
50	0.00	0.02	1.00	1.00	0.77	0.77	0.86	0.14	0.20	0.18	0.89	1.21	0.95	0.01	0.05	0.97	0.02	**0.04**
51	0.01	0.86	1.31	0.97	0.92	0.92	0.85	0.66	**0.68**	1.00	1.00	1.00	0.94	0.98	0.98	0.81	1.00	1.02
52	0.00	0.68	1.21	0.40	1.00	1.17	0.63	0.79	0.87	0.84	1.00	1.01	0.73	1.00	1.04	0.74	0.55	**0.61**
53	0.06	0.82	1.25	0.79	1.00	1.02	0.69	1.00	1.05	0.88	1.00	1.01	0.84	0.09	**0.19**	0.85	1.00	1.01
54	0.00	0.20	1.02	1.00	0.51	0.51	0.84	0.01	0.16	0.50	0.00	0.50	0.94	0.01	0.06	1.00	0.02	**0.02**
55	0.39	0.75	0.96	0.86	1.00	1.01	0.59	0.32	0.52	0.49	0.00	0.51	0.76	0.44	0.50	0.71	0.00	**0.29**
56	0.52	0.45	0.65	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.38	**0.38**	0.98	1.00	1.00	0.81	1.00	1.02
57	0.36	0.09	0.65	0.96	0.92	0.92	1.00	0.67	0.67	1.00	0.21	0.21	0.99	0.01	**0.01**	0.95	1.00	1.00
58	0.00	0.05	1.00	0.97	0.80	0.80	0.62	0.00	0.38	0.18	0.00	0.82	0.83	0.00	0.17	0.88	0.00	**0.12**
59	0.00	1.00	1.41	1.00	1.00	1.00	0.73	0.79	0.83	0.53	0.00	0.47	0.44	0.00	0.56	0.61	0.01	**0.39**
Average	0.19	0.42	0.91	0.83	0.43	0.46	0.89	0.84	0.85	0.71	0.28	0.40	0.91	0.33	0.34	0.86	0.28	**0.31**

The analysis of the results presented in Table 5.1 shows that the proposed mcDTSR method outperforms the state-of-the-art algorithms in 43 of the 59 videos, while also pairing up in 4 other videos. This shows that mcDTSR has superior individual performance over the other algorithms, but also that using domain transformations to deal reference/target misalignments is an improvement over mcRoSuRe-A, which is also a sparse representation technique. The average results of Table I

can be seen in Table 5.2, which shows that our method significantly reduces the average DIS score. This confirms the effectiveness of the introduction of the domain transformations in the detection pipeline.

It can be argued that object-level detection is a very harsh metric in applicability terms, so one may consider a less strict metric for detection performance like a frame-level analysis, which does not consider the anomaly position in the given frame. In this frame-level context the TP_{fl} metric is affected by the presence of any blob detected in an anomalous frame. Conversely, the FP_{fl} is determined by the presence of any blob detected in a non-anomalous frame. The average results for the frame-level comparison over the same 59 VDAO-200 videos is summarized in Table 5.3. These results confirm that mcDTSR has a clear performance advantage when compared to the STC-mc, DAOMC, MCBS, and ADMULT methods, and, more importantly, it represents a significant detection improvement when compared directly to the mcRoSuRe-A, even in a frame-level analysis.

Table 5.2: Average detection of proposed mcDTSR method compared to mcRoSuRe-A, STC-mc, DAOMC, and MCBS methods for all 59 single-object videos of the VDAO database.

Method	TP	FP	DIS
STC-mc	0.18	0.38	0.90
DAOMC	0.83	0.43	0.46
MCBS	0.89	0.84	0.85
mcRoSuRe-A	0.72	0.25	0.37
mcDTSR	0.84	0.27	**0.31**

Table 5.3: Average detection of proposed mcDTSR method, compared to STC-mc, DAOMC, MCBS, and mcRoSuRe-A methods for all 59 single-object videos of the VDAO database using frame-level metrics.

Method	TP_{fl}	FP_{fl}	DIS_{fl}
STC-mc	0.48	0.41	0.66
DAOMC	0.89	0.46	0.47
MCBS	0.99	0.98	0.98
mcRoSuRe-A	0.76	0.24	0.34
mcDTSR	0.86	0.23	**0.27**

The post-processing setups for the mcDTSR and mcRoSuRe-A algorithms, used to obtain the results shown in Tables 5.1, 5.2 and 5.3, were adjusted by a simple methodology. A parameter gridsearch is performed on 28 of the 59 videos, in order to minimize the object-level DIS score. The best parameter setup is then used to compute the results for all 59 videos in all three tables. In that search, the morphological open and close windows range, each one, from 1 to 5; the threshold

ranges from 0.2 to 0.3 in 0.01-steps, and the temporal voting window are tested for the 3, 5, and 7 sizes. Although the same tuning methodology was used for both algorithms, the best setup is chosen independently from each other, yielding the best average results of the object-level DIS metric for each method. This explains why the FP score of mcRoSuRe in Table 5.2 is slightly better than the one for the mcDTSR, something that is compensated by a large margin by the superior mcDTSR TP score. The optimal setup for each algorithm is shown in Table IV. In practice, one may expect that the proposed mcDTSR will behave equally or superior in terms of overall performance, as well aligned cases shall provide similar scores, but heavily misaligned videos will benefit from the domain transformation present in the mcDTSR algorithm.

Table 5.4: Chosen setup for methods mcRoSuRe-A and mcDTSR.

Method	Open Size	Close Size	Binarization Threshold	Vote Size
mcRoSuRe-A	4	1	0.20	3
mcDTSR	1	4	0.28	3

Fig. 5.7 illustrates the superiority of mcDTSR relative to mcRoSuRe-A in the presence a significant geometric misalignment between the target and reference frames, practically eliminating the false-alarm regions on the final residue matrix. This is one of the many examples where all the previous methods fail (having DIS metric larger than 0.85) and the proposed method excels, having a DIS metric of only 0.03.

Since the mcDTSR is based on the same sparse representation algorithm present in mcRoSuRe-A, the mcDTSR exhibits lower FP rates, in general, due to its intrinsic compensation of the camera trepidations. However, there are some situations in Table 5.1 where the mcDTSR presents more false alarms than mcRoSuRe. Having a close look at some of these cases, it is possible to see that mcDTSR captures not only the presence of abandoned objects, but also indirect visual artifacts caused by them, such as shadows and reflections, which were not considered in the ground-truth annotation for the VDAO database. This is illustrated in Fig. 5.8, which shows how the mcDTSR captures the shadow that the box casts in the lower pipe (Fig. 5.8(d)) yielding a false alarm region, which is ignored by mcRoSuRe-A method (Fig. 5.8(c)). Indeed, in most practical applications, this behavior, besides not being an issue, is even desirable. This is so because the goal is to find abandoned objects or anomalies, and therefore the detection of indirect artifacts caused by them is useful.

It is important to point out that the mcDTSR algorithm performs a sequence of convex optimizations in order to correct the geometric differences between the

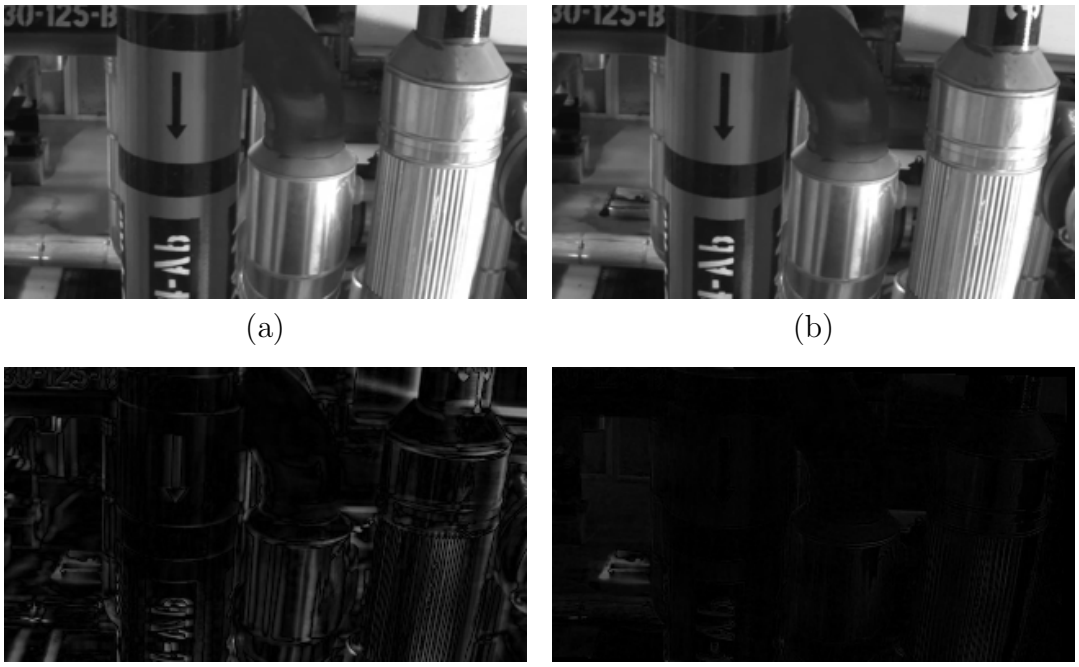
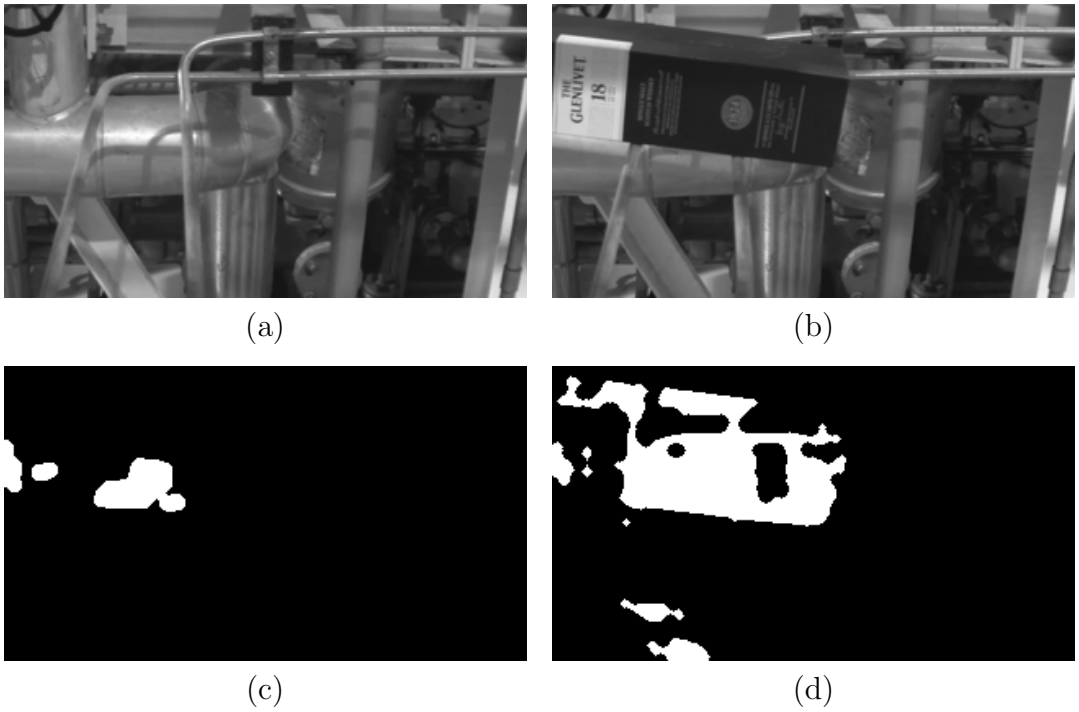


Figure 5.7: Comparison of mcRoSuRe and mcDTSR residues for frame 150 of video 41 in VDAO-200 database: (a) Reference frame; (b) Target frame; (c) mcRoSuRe-A residue $|E|$; (d) mcDTSR residue $|E'|$. In this case, the mcDTSR method compensates for frame misalignment removing most of false alarms regions.

reference and target domains. In practice, a more misaligned case tends to take more steps to yield the correct alignment, and consequently, requires more processing time. Although real-time performance was not in the scope of this work, it is important for real-world applicability. One form to address this issue is to develop an accelerated version of mcDTSR along the same lines that mcRoSuRe-A, an accelerated version of mcRoSuRe, has been developed. Further acceleration of mcDTSR can be provided by taking advantage of the expected sparsity of the W matrix, which can reduce significantly the amount of computation in the optimization loops.



(a) (b)

(c) (d)

Figure 5.8: Comparison of mcRoSuRe and mcDTSR results for frame 1 of video 1 in VDAO-200 database: (a) Reference frame; (b) Target frame; (c) mcRoSuRe-A detection mask; (d) mcDTSR detection mask. In this case, the shadow cast by the box, which does not have a bounding-box ground-truth counterpart, is ignored by the mcRoSuRe-A method but is successfully detected by the mcDTSR algorithm.

Chapter 6

Conclusions

6.1 Discussion

We have presented a new approach for detecting changes in videos acquired with a moving camera by applying special matrix decompositions with sparse low-rank representations. In the first method, the moving background is modeled as a union of subspaces plus a sparse residual in a first stage, which are used to decompose any given target sequences using this same representation. The residual of this target representation is then subtracted from the reference residual, leaving only the detected outliers. In other words, this technique provides a moving background subtraction scheme that could be used in advanced surveillance tasks which require moving cameras to cover the surveilled area. This job is done with no previous geometric registration or any pose or position information from the camera or its moving platform, needing only a very loose video synchronization between reference and target videos.

Furthemore, a new algorithm that is tolerant to possible camera trepidations was described. The proposed method is based on the low-rank/sparse representation of target videos using a corresponding similar decomposition performed on an anomaly-free reference video. Both video representations are performed in a geometrically-transformed domain in order to compensate possible camera trepidations along its natural path. An iterative two-stage optimization procedure is employed to implement the modified optimization problem: the inner loop estimates the best geometric transformation, whereas the outer loop, given the current transformation estimate, determines the best matrix factorization. This provides a better registration between reference and target videos, reducing the amount of false alarms in the subsequent detection stage, which becomes robust to camera trepidations. The final algorithm has proven to be a powerful tool, in a way it is able to outperform state-of-the-art methods in the detection of anomalies in videos

acquired with a moving camera, as the results of the extensive experiments in a very challenging dataset show.

The proposed method expands the capabilities of low-rank sparse representation methods, such as mcRoSuRe-A, by incorporating, in a simple and elegant way, domain transformations that enable such methods to find more precise correspondences between different parts of the data matrix. The better alignment of the frames from reference and target videos make the experimental results of our method to present improved true positive detection while having very few false detections.

The optimization proposed here takes inspiration on other well established domain transformation techniques, but goes further using convexification tools that reach faster convergence, which allows the method to operate in challenging scenarios.

Although other recent works have proposed the use of domain transformations to improve background subtraction methods, none of the previous algorithms was able to perform in the challenging scenario considered here. This opens a path for new applications in trending areas such as video stabilization and anomaly detection with freely moving cameras that currently lack simple tools that can be incorporated into the optimization algorithm to handle large misalignment between frames.

6.2 Future Work

The proposed technique can be improved in various aspects in order to become a complete system of change detection that works in real-time. Future implementations should take advantage of the expected sparsity of output matrices. As can be seen in Figure 5.6, the resulting coefficient matrix W, as defined in Eq. (5.15), has only a few nonzero entries. This fact can be exploited to create accelerated implementations of Algorithms 6 and 7 by avoiding massive unnecessary computation. It is also important to take into account the evolution of matrix W through the outer loop iterations and how the nonzero entries "move" along W at each step (see Figure 5.6).

Another important concern is that, since current camera resolutions continue to grow, working with huge data, as the videos available in the VDAO database, might not be a feasible approach. Instead, a multiscale divide-and-conquer strategy should be more efficient, both in terms of time and data consumption. Compressive sensing theory applied to the matrix completion problem has shown that low-rank matrices can be exactly reconstructed from very little retrieved information [21–24]. Exploiting this fact, an interesting alternative is to use subsampled versions of the video tensor, which represent a rough scale of the information. An interesting and useful question is how much one can shrink the data and still detect the changes

in the target videos. In addition, better quality metrics and evaluation strategies should be proposed.

Subsampling of the data will enable the processing of longer videos. It means that it may be possible to process slices of the VDAO videos that may contain at least one full lap of the robot. Working with these kind of reference videos may raise the following question: is it possible to estimate L_r from equation (4.1) with fewer samples than the reference matrix X_r? This can be approached as a dictionary learning problem [49, 50] and one should investigate how compact this dictionary could be. Another interesting strategy is that L_r could be divided into small clusters to take advantage of its blockwise low-rank structure. Subspace clustering in low-rank representation is discussed in several works [40–43]. This approach could be useful for parallelization strategies to scale the method to work with larger data.

Finally, the robustness of the complete technique should be assessed to ensure its utility and confidence. Several experiments should be planned and executed to understand its efficacy and limitations. Other video datasets and real time implementations may be considered as optional goals.

Bibliography

[1] R. J. Radke, S. Andra, O. Al-Kofahi, and B. Roysam, "Image change detection algorithms: A systematic survey," *IEEE Trans. Image Process.*, vol. 14, no. 3, pp. 294-307, Mar. 2005.

[2] Y. Tomioka, A. Takara, and H. Kitazawa, "Generation of an optimum patrol course for mobile surveillance camera", *Transactions on Circuits and Systems for Video Technology*, vol. 2, no. 22, pp. 216-224, Feb. 2012.

[3] P. Gupta, G. N. Purohit, and A. Dadhich, "Crime Prevention through Alternate Route Finding In Traffic Surveillance Using CCTV Cameras", *International Journal of Engineering and Advanced Technology*, vol. 2, no. 5, pp. 414–418, 2013.

[4] Y. Sheikh, O. Javed, and T. Kanade, "Background subtraction for freely moving cameras," *Proc. IEEE Int. Conf. Comput. Vision*, pp. 1219-1225, 2009.

[5] H. Kong, J.-Y. Audibert, and J. Ponce, "Detecting abandoned objects with a moving camera," *IEEE Trans. Image Processing*, vol. 19, no. 8, pp. 2201-2210, Aug. 2010.

[6] O. M. Sincan, V. B. Ajabshir, H. Y. Keles, and S. Tosun, "Moving object detection by a mounted moving camera," *IEEE Int. Conference on Comput. as a Tool (EUROCON)*, pp. 1-6, Sep. 2015.

[7] V. Chandola, A. Banerjee, and V. Kumar. "Anomaly detection: A survey," *ACM Comput. Surv.* vol. 41, no. 3, art. 15, pp. 1–58, Jul. 2009.

[8] X. Cui, J. Huang, S. Zhang, and D. N. Metaxas, "Background subtraction using low rank and group sparsity constraints," *Proc. European Conf. Comp. Vision (ECCV)*, Part I, LNCS 7572, pp. 612-625, Oct. 2012.

[9] K. Toyama, J. Krumm, B., Brumitt, and B. Meyers, "Wallflower: principles and practice of background maintenance," *Int. Conference on Comput. Vision*, pp. 255-261, 1999

[10] L. Li, W. Huang, I. Gu, and Q. Tian, "Statistical modeling of complex backgrounds for foreground object detection," *IEEE Transactions on Image Processing*, vol. 13, no. 11, pp. 1459-1472, 2004.

[11] R. Tron, and R. Vidal, "A benchmark for the comparison of 3-d motion segmentation algorithms," *IEEE International Conference on Computer Vision and Pattern Recognition*, Jun. 2007

[12] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, "ChangeDetection.NET: A new change detection benchmark dataset," *in Proc. IEEE Workshop on Change Detection (CDW-2012) at CVPR-2012*, Providence, Jun. 2012.

[13] CDNET: ChangeDetection.NET: 2012 and 2014 databases [online]. Available: http://changedetection.net or http://wordpress-jodoin.dmi.usherb.ca

[14] A. F. da Silva, L. A. Thomaz, G. Carvalho, M. T. Nakahata, E. Jardim, J. F. L. de Oliveira, E. A. B. da Silva, S. L. Netto, G. Freitas, and R. R. Costa, "An annotated video database for abandoned-object detection in a cluttered environment," *Proc. Int. Telecommun. Symp. (ITS)*, São Paulo, Brazil, Aug. 2014.

[15] VDAO: Video database of abandoned objects in a cluttered industrial environment [online]. Available: http://www.smt.ufrj.br/∼tvdigital/database/objects

[16] VDAO-200: 200-frame excerpts form VDAO database [online]. Available at: http://www02.smt.ufrj.br/∼tvdigital/database/research

[17] C. Cuevas, R. Martínez, and N. García, "Detection of stationary foreground objects: A survey," Computer Vision and Image Understanding, vol. 152, pp. 41–57, Nov. 2016.

[18] C. Guyon, T. Bouwmans and E.-H. Zahzah, "Robust principal component analysis for background subtraction: Systematic evaluation and comparative analysis," in *Principal Component Analysis, Dr. Parinya Sanguansat (Ed.)*, ISBN: 978-953-51-0195-6, InTech, 2012. Available from: http://www.intechopen.com/books/principal-component-analysis/robust-principal-component-analysis-for-background-subtraction-systematic-evaluation-and-comparative

[19] M. Turk and A. Pentland, "Eigenfaces for Recognition," *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71-86, 1991.

[20] R. Basri and D. Jacobs, "Lambertian reflectance and linear subspaces," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 25, no. 2, pp. 218-233, 2003.

[21] E. J. Candès and B. Recht, "Exact matrix completion via convex optimzation," *Found. of Comput. Math*, no. 9, pp. 717-772, 2009.

[22] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?," arXiv:0912.3599v1, Dec. 2009

[23] J. Wright, Y. Peng, Y. Ma, A. Ganesh, and S. Rao, "Robust principal component analysis: Exact recovery of corrupted low-rank matrices by convex optimization," *Neural Information Processing Systems (NIPS*, 2009.

[24] Z. Zhou, X. Li, J. Wright, E. Candès, and Y. Ma, "Stable principal component pursuit," arXiv:1001.2363v1, Jan. 2010

[25] A. Y. Yang, Z. Zhou, A. G. Balasubramanian, S. S. Sastry, and Y. Ma, "Fast l_1-minimization algorithms for robust face recognition," *IEEE Trans. Image Processing*, vol. 22, no. 8, pp. 3234-3246, Aug. 2013.

[26] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 210-227, Feb. 2009.

[27] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T. Huang, and S. Yan, "Sparse representation for computer vision and pattern recognition," *Proc. IEEE*, vol. 98, no. 6, pp. 1031-1044, Jun. 2010.

[28] A. Wagner, J. Wright, A. Ganesh, Z. Zhou, H. Mobahi, and Y. Ma, "Toward a practical face recognition system: Robust alignment and illumination by sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 2, pp. 372-386, Feb. 2012.

[29] X. Zhou, C. Yang, and W. Yu, "Moving object detection by by detecting contiguous outliers in the low-rank representation," *IEEE Trans. on Pattern Anal. and Mach. Intell. (PAMI)*, Mar. 2013.

[30] Z. Lin, R. Liu, and Z. Su, "Linearized Alternating Direction Method with Adaptive Penalty for Low-Rank Representation," *Advances in Neural Information Processing Systems 24*, pp. 612-620, 2011.

[31] G. Carvalho, "Automatic detection of abandoned objects with a moving camera using multiscale video analysis," *Ph.D. dissertation*, PEE, UFRJ, Rio de Janeiro, RJ, 2015.

[32] X. Bian and H. Krim, "Bi-sparsity pursuit for robust subspace recovery," in *IEEE International Conference on Image Processing*, Quebec, Canada, September 2015, pp. 3535–3539.

[33] X. Bian and H. Krim, "Robust subspace recovery via bi-sparsity pursuit," arXiv:1403.8067v2, Apr. 2014.

[34] E. Jardim, X. Bian, E. A. B. da Silva, S. L. Netto, and H. Krim, "On the detection of abandoned objects with a moving camera using robust subspace recovery and sparse representation," *IEEE Int. Conference on Acoust., Speech, and Signal Process. (ICASSP)*, Brisbane. pp. 1295-1299, Apr. 2015

[35] L. A. Thomaz, A. F. da Silva, E. A. B. da Silva, S. L. Netto, X. Bian and H. Krim, "Abandoned object detection using operator-space pursuit," *IEEE Int. Conference on Image Process. (ICIP)*, Quebec City. v. 2, pp. 1980-1984, Sep. 2015

[36] L. A. Thomaz, E. Jardim, A. F. da Silva, E. A. B. da Silva, S. L. Netto, and H. Krim, "Anomaly detection in moving-camera video sequences using principal subspace analysis", *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 65, no. 3, pp. 1003–1015, Mar. 2018.

[37] M. T. Nakahata, L. A. Thomaz, A. F. da Silva, E. A. B. da Silva, and S. L. Netto, "Anomaly detection with a moving camera using spatio-temporal codebooks", *Multidimensional Systems and Signal Processing*, vol. 24, no. 3, pp. 1025-1054, July 2018.

[38] H. Mukojima, D. Deguchi, Y. Kawanishi, I. Ide, H. Murase, M. Ukai, N. Nagamine, and R. Nakasone, "Moving camera background-subtraction for obstacle detection on railway tracks," *Proc. IEEE International Conference on Image Processing*, Phoenix, USA, pp. 3967–3971, Sept. 2016.

[39] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, 2004.

[40] E. Elhamifar, and R. Vidal. "Sparse subspace clustering," in *IEEE Conference on Comput. Vision and Pattern Recognition (CVPR)*, pp. 2790-2797, 2009.

[41] E. Elhamifar, R. Vidal, "Sparse subspace clustering: Algorithm, theory, and applications," *IEEE Trans. on Pattern Anal. and Mach. Intell.*, vol. 35, no. 11, pp. 2765-2781, Nov. 2013

[42] M. Soltanolkotabi and E. J. Candès, "A geometric analysis of subspace clustering with outliers," *The Annals of Statistics*, vol. 40, no. 4, pp. 2195-2238, 2012.

[43] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, "Robust recovery of subspace structures by low-rank representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 171-184, Jan. 2013.

[44] Z. Zhang, A. Ganesh, X. Liang, and Y. Ma. "TILT: Transform invariant low-rank textures," *in 10th Asian Conference on Computer Vision (ACCV)*, Nov. 2010.

[45] Z. Zhang, A. Ganesh, X. Liang, and Y. Ma. "TILT: Transform invariant low-rank textures," *Int. Journal of Computer Vision (IJCV)*, vol. 99 no. 1, pp. 1-24, 2011.

[46] Y. Peng, A. Ganesh, J. Wright, W. Xu, and Y. Ma, "RASL: Robust alignment via sparse and low-rank decomposition for linearly coorelated images," in Proc. of *IEEE Int. Conference on Comput. Vision and Pattern Recognition (CVPR)*, pp. 763-770, 2010.

[47] Y. Peng, A. Ganesh, J. Wright, W. Xu, and Y. Ma, "RASL: Robust alignment via sparse and low-rank decomposition for linearly coorelated images," *IEEE Trans. on Pattern Anal. and Mach. Intell. (PAMI)*, vol. 34, no. 11, pp. 2233-2246, Jul. 2010.

[48] M. A. Fischler, R. C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Comm. of the ACM*, vol. 24, pp. 381-395, Jun-1981.

[49] E. Elhamifar, G. Sapiro, and R. Vidal. "See all by looking at a few: Sparse modeling for finding representative objects," in *IEEE Conference on Comput. Vision and Pattern Recognition (CVPR)*, pp. 1600-1607, 2012.

[50] X. Bian, H. Krim, A. Bronstein, and L. Dai, "Sparse null space basis pursuit and analysis dictionary learning for high-dimensional data analysis," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015

[51] J. He, D. Zhang, L. Balzano, and T. Tao, "Iterative Grassmannian optimization for robust image alignment," *Image and Vision Computing*, vol. 32, no. 10, pp. 800–813, 2014.

[52] H. Yong, D. Meng, W. Zuo, and L. Zhang, "Robust online matrix factorization for dynamic background subtraction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 7, pp. 1726–1740, 2018.

[53] Y. Wu, B. Shen, and H. Ling, "Online robust image alignment via iterative convex optimization," *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Providence, USA, pp. 1808–1814, June 2012.