Pattern Recognition 42 (2009) 1485 – 1495

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

A fusion-based approach to digital movie restoration

Lucia Maddalena[a,*], Alfredo Petrosino[b,1], Giuliano Laccetti[c,2]

[a]*National Research Council, Institute for High-Performance Computing and Networking, Via P. Castellino 111, 80131 Naples, Italy*
[b]*University of Naples Parthenope, Department of Applied Science, Centro Direzionale Isola C/4, 80143 Naples, Italy*
[c]*University of Naples Federico II, Department of Mathematics and Applications, Via Cintia, 80126 Naples, Italy*

ARTICLE INFO

Article history:
Received 22 July 2007
Received in revised form 9 October 2008
Accepted 23 October 2008

Keywords:
Digital film restoration
Data fusion techniques
Parallel computation

ABSTRACT

Many algorithms have been proposed in literature for digital movie restoration; unfortunately, none of them ensures a perfect result whichever is the image sequence to be restored. Here we propose a new digital scratch restoration algorithm which achieves accuracy results higher than that of already existing algorithms and naturally adapts for implementation into high-performance computing environments.
The basic idea of the proposed algorithm is to adopt several relatively well-settled algorithms for the problem at hand and combine obtained results through suitable image fusion techniques, with the aim of taking advantage of the adopted algorithms' capabilities and, at the same time, limiting their deficiencies. Extensive experiments on real image sequences deeply investigate both accuracy results of the presented scratch restoration approach, which is shown to outperform other existing approaches, and performance of its parallel implementation, which allows for real-time restoration.

1. Introduction

With the recent advent of digital technologies and the ever increasing need for speed and storage, occluded or missing parts in images and movies is a more and more widespread problem. The problem can occur in several multimedia applications, such as wireless communication and digital movie restoration, and can deeply affect also the performance of compression techniques. The specific application we are concerned with is the digital movie restoration, which is the set of image processing methodologies and techniques allowing to reproduce digital copies of damaged movies that are as much as possible similar to the uncorrupted lost original. Several classes of defects can be distinguished that affect movies; in the present paper, we focus on the class of scratch defects, intended as long and thin vertical scratches that affect several subsequent images of a sequence, due to the abrasion of the film by dust particles in the slippage mechanisms used for film development, projection and duplication.

Many scratch restoration methods are reported in literature; as expected, advantages and disadvantages characterize each scratch restoration technique, and any of them could be said to win the competition. We propose to deal with this kind of problems through the adoption of *fusion techniques*, usually adopted in pattern recognition. Indeed, it is now widely accepted that combining multiple classifiers can provide advantages over the traditional monolithic approach to pattern classifier design and many experimental works have shown the improvement in performance that can be achieved by multiple classifiers in several applications [1]. Some results concerning image analysis can be found in literature [2–6]. The innovative aspect of the research reported in this paper consists in considering images obtained by different restoration methods as providers of alternative and complementary "views" and "characteristics" of a given area. We propose to suitably "fuse" them into a single image, i.e. the final restored image, so that all the important visual information found in the input images is transferred into the fused output image, without the introduction of artifact distortion effects. In this sense machine vision systems can be organized as a set of separated visual modules that act as virtual sensors. In this paper, the term *visual module* indicates an algorithm that extracts information of some specific and descriptive kind from a digital image.

In the context of digital scratch restoration, fusion techniques may be applied to both the detection and the removal stages of the automatic restoration process (see Section 2). Here we propose a new approach to line scratch restoration based on fusion techniques which for both the stages takes into account already existing promising algorithms and suitably combines, through existing and new image fusion techniques, the obtained results in order to provide a restored sequence as similar as possible to the original

* Corresponding author. Tel.: +39 081 6139522; fax: +39 081 6139531.
E-mail addresses: lucia.maddalena@na.icar.cnr.it (L. Maddalena), alfredo.petrosino@uniparthenope.it (A. Petrosino), giuliano.laccetti@dma.unina.it (G. Laccetti).
1 Tel.: +39 081 5476601; fax: +39 081 5476514.
2 Tel.: +39 08 1675619; fax: +39 08 1675636.

0031-3203/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2008.10.026

uncorrupted sequence. As we show, the results produced by the proposed approach over different damaged movies greatly enhance those produced separately by each considered method.

On the other side, automatic batch restoration is limited to low degraded video sections and is mainly aimed at the removal of speckle noise and brightness variations. The manual inpainting of high definition digital video is, of course, a very intensive and expensive activity which can be justified in cases of documents of particular historical and artistic meaningfulness. A great improvement in this field should be related to the development of high-performance software for the automatic or semiautomatic restoration of degraded image sequences; this could drastically reduce the costs and efforts for the restoration of whole movies, at the same time allowing the use of highly specialized algorithms.

We show that the proposed scratch restoration algorithm is naturally suited for implementation into high-performance computing environments, presenting a parallel version of the proposed approach, based on both task and data partitioning strategies, and analyzing its performance on real image sequences.

The contents of this paper are as follows: in Section 2 we describe the scratch restoration problem we are concerned with, outlining the usual restoration process and giving a brief overview of methods reported in literature. In Section 3 we outline the proposed fusion-based algorithm, giving a brief description of the adopted underlying restoration algorithms and fusion strategies, and we present the parallelization strategy and a performance model for the parallel algorithm. In Section 4 we describe accuracy results achieved by the proposed approach tested on real image sequences and present experimental results concerning parallel execution times and speedup. Conclusions are reported in Section 5.

2. Line scratch restoration: the problem

A sufficiently general model of degraded video signal is the following for a pixel location $\mathbf{p} = (x, y)$:

$$I(\mathbf{p}, t) = (1 - b(\mathbf{p}, t))I^*(\mathbf{p}, t) + b(\mathbf{p}, t)c(\mathbf{p}, t) + \eta(\mathbf{p}, t) \qquad (1)$$

where $I(\mathbf{p}, t)$ is the corrupted signal at spatial position $\mathbf{p}$ in frame t, $b(\mathbf{p}, t) \in \{0, 1\}$ is a binary mask indicating the points belonging to missing parts of the degraded video, I^* is the ideal uncorrupted image. The (more or less constant) intensity values at the corrupted spatial locations are given by $c(\mathbf{p}, t)$. Though noise is not considered to be the dominant degrading factor in the defect domain, it is still included in Eq. (1) as the term $\eta(\mathbf{p}, t)$.

In the present paper we focus on the class of line scratch defects, intended as long and thin vertical scratches that affect several subsequent images of a sequence, due to the abrasion of the film by dust particles in the slippage mechanisms used for the development, projection and duplication of the film.

Commonly, scratch restoration is a two-step procedure. In the first step the scratches need to be detected, i.e. an estimate for the mask $b(\mathbf{p}, t)$ is made (*detection step*). In the second step the values of I^* inside the scratch, possibly starting from information about $c(\mathbf{p}, t)$, are estimated (*removal step*).

Several scratch restoration methods are reported in the literature, and many others can be derived from methods for the more general problem of missing data restoration.

The detection step is usually accomplished through low-/high-pass filters [7], morphological filters [8], adaptive binarization [9], statistics and MAP techniques [10], eventually coupled with techniques such as Hough transform [7] or Kalman filter [8], and possibly followed by Bayesian refinement strategies [7].

The removal step is obtained through interpolation or approximation [10,11], the adoption of autoregressive models [7] or mean vector filters [9], eventually followed by the reconstruction of the image grain (high-frequency components) via Fourier series [8] or least squares-based grain estimation [11]. Scratch removal techniques can also be borrowed by *image inpainting*, that is the set of techniques for making undetectable modifications to images [12]. Such techniques are generally used to fill-in missing data or to substitute information contained in small image regions [13]. Inpainting is also related to *texture synthesis*, where the problem consists in generating, given a sample texture, an unlimited amount of image data which will be perceived by humans as having the same texture [14–16].

Even though many of the cited algorithms achieve very accurate results, none of them ensures a perfect restoration whichever is the image sequence to be restored and whichever is the kind of scratch by which it is affected; therefore scratch restoration is still considered an open problem.

3. Line scratch restoration: the proposed algorithm

The proposed algorithm for scratch restoration in image sequences is based on a new methodology for the solution to classes of problems to be dealt with in digital movie restoration. The algorithm takes into account already existing promising algorithms and suitably combines by fusion the obtained results in order to provide a restored sequence as similar as possible to the original uncorrupted sequence, enhancing [17] in terms of proposed fusion techniques.

With *fusion techniques* input images which provide alternative and complementary "views" and "characteristics" of a given area are "fused" into a single image (see for instance [2–6]). Fusion techniques should ensure that all the important visual information found in input images is transferred into the fused output image, without the introduction of artifact distortion effects.

Concerning digital scratch restoration, fusion techniques may be applied both to the detection stage and to the removal stage of the automatic restoration process. Therefore, the basic idea of the *fusion-based algorithm*, which we call *LSR algorithm*, consists, for each sequence frame, in:

(1) applying a set of d existing scratch detection algorithms;
(2) combining obtained scratch masks $B^j, j = 1, \ldots, d$, to produce the final scratch mask B^C;
(3) applying a set of r existing scratch removal algorithms using scratch mask B^C;
(4) combining obtained restored images $R^j, j = 1, \ldots, r$, to produce the final restored image R^C.

The algorithm is thus parametrized by: (a) the number d and typology of scratch detection algorithms; (b) the combination rule adopted for scratch detection stage; (c) the number r and typology of scratch removal algorithms and (d) the combination rule adopted for scratch removal stage. Concerning items (a) and (c), a possible scheme of the proposed LSR algorithm is given in Fig. 1, where $d = 3$ detection algorithms and $r = 2$ removal algorithms, briefly described in Sections 3.1 and 3.2, have been adopted. The number of restoration algorithms has been chosen for illustrative purposes, but also after having considered different combinations; moreover, such modules have been chosen on the basis of their proven effectiveness and in that they cover a broad range of approaches to restoration problems. For what concerns items (b) and (d), our choices will be described and motivated in Sections 3.1.4 and 3.2.3.

3.1. Fusion-based detection

In this section we consider the problem of scratch detection, that is, given the corrupted image sequence I, construct an estimate B for the mask b in Eq. (1). Our approach is to use more than one

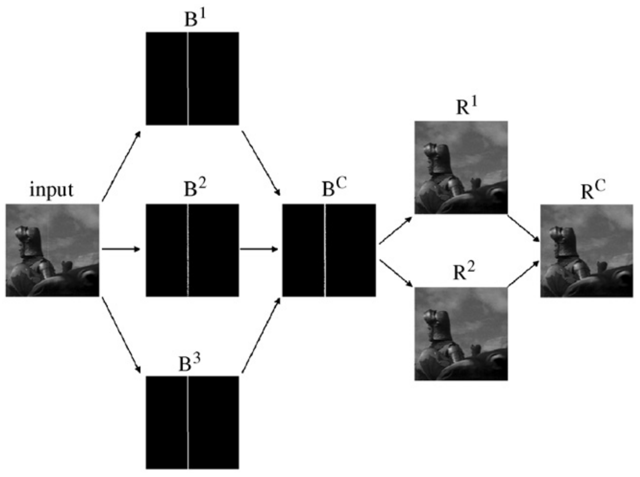


Fig. 1. A scheme of the LSR algorithm.

detection visual module in order to make up for deficiencies in the individual modules, while retaining their features, thus achieving a better overall detection result than each single module could provide.

In the following some hints will be provided about the chosen visual modules, that perform detection of scratches of width w in sequences of images of height N and width M, and about different fusion techniques we tailored to handle the problem at hand.

3.1.1. Detection Algorithm 1

Following [8], the first detection algorithm taken into account can be described as follows:

- pre-processing of each sequence frame, obtained by using the Radon projection in the vertical direction, with height S depending on the maximum inclination of the scratch to the vertical direction and on the image spatial resolution:

$$I_S(x,y) = \sum_{j=0}^{S-1} I(x, j + S * y)$$

producing an image I_S with height N/S;
- vertical linear structures detection in subsampled images I_S by a local maximum detection procedure based on a top-hat morphological transformation, with structuring element of dimension $w+1$, followed by a thresholding process and by the line Hough transform;
- creation of the scratch mask B^1 for each sequence frame, using the results obtained in the previous phase on subsampled images I_S.

The produced results could be enhanced, by retaining only the scratches that appear fixed in time; as suggested in [8] this can be made by tracking the scratches. Anyway to maintain comparable the computational complexities of the considered detection algorithms, we did not implement this part of the procedure.

3.1.2. Detection Algorithm 2

In [9] the detection scheme consists of:

- adaptive binarization of each sequence frame, producing a binary image C, by adopting a threshold σ computed over a horizontal neighborhood of any pixel at position (x, y) as

$$\sigma = \frac{1}{n} \sum_{i=-n/2}^{n/2} I(x+i, y)$$

where $n = w+1$ for even w and $n = w+2$ for odd w;
- construction of the scratch mask B^2 by labeling as belonging to the scratch all the pixels of columns x of C that contain at least 70% of pixels with value $C(x, \cdot) = 1$.

3.1.3. Detection Algorithm 3

Following [7], the third detection algorithm taken into account consists of:

- low-pass filtering with a Gaussian vertical filter, followed by a vertical subsampling of width S, producing an image I_S with height N/S;
- location of the horizontally impulsive features in I_S by:
  - horizontal median filtering I_S, in order to eliminate straight lines with width lower than the filter window size $2w+1$, producing I_M;
  - thresholding the difference $I_S - I_M$, in order to obtain the binary mask C;
  - line Hough transform in order to characterize the straight lines in C (the candidate scratches);
  - creation of the scratch mask B^3 for each sequence frame, using the results obtained in the previous phase on subsampled images I_S.

In [7] this process is followed by a Bayesian refinement strategy which allows to select, among candidate scratches, those that can be best modeled by a horizontal scratch section model. As for detection Algorithm 1, we avoided to implement this part of the procedure.

3.1.4. Detection fusion strategies

In this case the combination should be made among scratch masks B^j, $j = 1, \ldots, d$, produced by the d detection modules. Here, two different combining methods or aggregation operators are adopted: the *union aggregation operator* and a newly proposed technique, named *maximum covering (MC) aggregation operator*. Supposing, for simplicity, that damaged images are affected by just one scratch, their result B^C is given by

- *Union aggregation operator*: $B^C = \cup\{B^j : j = 1, \ldots, d\}$ such that

$$B^C(x,y) = \max_j\{B^j(x,y)\}, \quad \forall (x,y).$$

- *MC aggregation operator*: $B^C = MC\{B^j : j = 1, \ldots, d\}$ such that for all $y = 0, \ldots, N-1$ (N = number of image rows),

$$B^C(x,y) = \begin{cases} 1 & \text{if } x \in [\bar{x} - W, \bar{x} + W] \\ 0 & \text{otherwise} \end{cases}$$

where $W = \max\{|\bar{x} - x^{min}|, |\bar{x} - x^{max}|\}$, $\bar{x} = \text{mean}(X)$, $x^{min} = \min(X)$, $x^{max} = \max(X)$, $X = \{x : \cap_j B^j(x,y) = \min_j\{B^j(x,y)\} = 1, \forall y\}$.

The basic idea of the union aggregation operator is to include into the detection mask all pixels considered as corrupted by any of the underlying detection method, regardless their being true positives or false alarms, and to exclude any other pixel.

On the other side, the idea underlying the MC aggregation operator is to include into the detection mask only those pixels considered as corrupted by most of the underlying algorithms, excluding outliers. Therefore, the center column of the scratch is chosen as the mean center column of the scratch masks intersection ($\bar{x}$), while the scratch half width (W) is computed as the widest half width of the scratch masks intersection.

3.2. Fusion-based removal

This section explores the scratch removal problem, that is, given the corrupted image sequence I and the estimated scratch mask B, construct an estimate R of the ideal uncorrupted image I^* in Eq. (1).

The fusion approach can be here stated as follows: given r images representing heterogeneous data on the observed phenomenon, take a decision D_i on an element (x,y) where D_i belongs to a decision space D. In image fusion the information relating (x,y) to each possible decision D_i is represented as a number M_i^j, where j indexes the decision making module having different properties and different meanings depending on the mathematical framework.

Concerning single methods we consider one belonging to the class of interpolating methods [11] and the other one to the class of inpainting algorithms [16]. Both algorithms attempt to reconstruct not only the structure of the image in the scratch domain (pixels where mask B gets value 1), but also its texture.

3.2.1. Removal Algorithm 1

The first scratch removal algorithm taken into account is based on a non-parametric Markovian model adopted in [16] for texture synthesis, and adapted to our purposes. The probabilistic model is based on the assumption of spatial locality: the probability distribution for one pixel given the values of its neighborhood is independent of the rest of the image. The model is non-parametric in the sense that the probability function is not imposed or constructed explicitly, but it is approximated by a reference sample image, which must be large enough to capture the stationarity of the texture.

For each pixel p to be reconstructed, the algorithm proceeds as follows:

- determination of the sample image C, chosen as the square window of dimension A centered in p;
- construction of the set Q of pixels in C having a neighborhood *similar* to that of p. The similarity of two neighborhoods is measured according to the normalized sum of squared differences and it is weighted by a two-dimensional Gaussian of dimension G, in order to give more importance to the pixels that are near the center of the window than to those at the edge;
- reconstruction of pixel p, obtained assigning it the intensity value of a pixel randomly drawn from the set Q.

In order to establish the reconstruction order, for each scratch pixel detected in the binary mask B the number of its valid neighbors is enumerated; pixels are then replaced starting from the ones having the most valid neighbors. Scratches are thus simultaneously and progressively filled from the edges to the center of the scratch.

3.2.2. Removal Algorithm 2

In [11] a simple interpolating method is adopted and the interpolation result is corrected by adding to it the estimated displacement between the adopted model and the real model. Specifically, provided the scratch mask B, the procedure consists of:

- interpolation of the pixels pertaining to the scratch domain;
- estimate of the image texture in the scratch neighborhood, obtained by computing the displacement between the least square fitting over an uncorrupted neighborhood of the scratch and the same neighborhood pixels;
- addition of the estimated texture to the pixels belonging to the scratch domain.

Different versions of the above described method can be obtained by adopting different interpolation methods and neighborhood sizes.

3.2.3. Removal fusion strategies

Given images R^j obtained by removal module j, $j=1,\dots,r$, if we assume that $M_i^j(x,y)=R^j(x,y)$, with (x,y) in the scratch domain, represents the probability degree to which the pixel (x,y) could be seen as "restored" (i indexes the values of this appearance), we can claim all the advantages of the Bayesian framework relying in the variety of combination operators. Here we adopt the weighted averaging aggregation operator, known in the Bayesian framework as the generalized ensemble method (GEM) [18] for combining different classification modules, which has been demonstrated to significantly improve the classification performance of each single module: $R^C = GEM\{R^j : j=1,\dots,r\}$, such that

$$R^C(x,y) = \frac{1}{r}\sum_{j=1}^{r} \alpha_j R^j(x,y) \quad \forall (x,y)$$

with $\alpha_j \in [0,1], j=1,\dots,r$, and $\sum_{j=1}^{r}\alpha_j = 1$. Weights α_j are chosen so as to minimize the mean square error (MSE) with respect to the target function $I^*(x,y)$ (the original uncorrupted image from Eq. (1)). If $I^*(x,y)$ is known, a closed form solution for α_j's can be obtained [18]. Specifically, in our case where $r=2$ we have $\forall(x,y)$

$$R^C(x,y) = \alpha_1 R^1(x,y) + \alpha_2 R^2(x,y) = \alpha R^1(x,y) + (1-\alpha)R^2(x,y) \tag{2}$$

and the weight α is the value which minimizes:

$$MSE[R^C] = f(\alpha) = \alpha^2 C_{1,1} + 2\alpha(1-\alpha)C_{1,2} + (1-\alpha)^2 C_{2,2}$$

where $C_{i,j}=E[m_i(x,y)m_j(x,y)], i,j=1,2$ are the coefficients of the symmetric correlation matrix, and $m_i(x,y)=I^*(x,y)-R^i(x,y), i=1,2$, represent the *misfit* of functions $R^i(x,y)$ (the deviation from the true solution). Setting to zero the derivative of $f(\alpha)$ with respect to α, we conclude that the minimum of $f(\alpha)$ is given by

$$\alpha = \frac{C_{2,2}-C_{1,2}}{C_{1,1}-2C_{1,2}+C_{2,2}} \tag{3}$$

When the target function $I^*(x,y)$ is unknown, choice of weights α_j should be driven by a priori information concerning the accuracy of the adopted removal modules. An analysis about the choice of weights α_j in our specific case is reported together with experiments at the end of Section 4.2.2.

Furthermore, we investigated the multiresolution (MR) image fusion approach, whose basic idea is to perform an MR transform on each source image and, following some specific fusion rules, construct a composite MR representation from these inputs; the composite final image is obtained by applying the inverse transform on this composite MR representation. However, experiments carried out using different MR schemes and different fusion rules provided in [19,20] showed that the adoption of such approach instead of the GEM aggregation operator does not improve removal accuracy of LSR algorithm.

3.3. Parallel restoration algorithm

Based on the consideration that detection modules $D_1,\dots,D_d$ are completely independent of each other, while the combining module for detection DC can be used only after the detection modules have been executed (the same is valid for removal modules $R_1,\dots,R_r$ and combining removal module RC), we devise an efficient parallel version of the LSR algorithm. The algorithm, named in the following *P-LSR algorithm*, adopts the dependency graph of the various modules that constitute the LSR algorithm, described through the image processing application task graph (IATG) shown in Fig. 2 [21]. We recall that an IATG can be modeled by a macro dataflow communication graph (MDG) [22], which is a directed acyclic graph where each node stands for a visual module and each edge stands for a precedence constraint between two adjacent operators. The given description of the LSR algorithm through the IATG of Fig. 2 easily allows to individuate a natural parallelization strategy based on task partitioning.

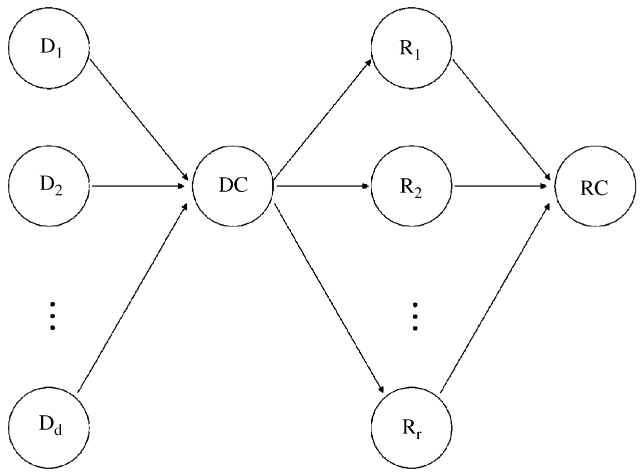


Fig. 2. The IATG for the LSR algorithm.

Each detection/removal/combination module can be seen as an individual task; however, they can have very different computational complexities (for example, removal Algorithm 1 takes more than 50% of total restoration time), and this task partitioning would result in load unbalance of the fusion-based algorithm. We therefore propose to consider as tasks: (a) the union of the first and the second graph columns (detection task) and (b) the union of the third and the fourth graph columns (removal tasks), such that each process is assigned either one of the two tasks.

In order to enhance the parallel algorithm scalability against the small number of tasks and the huge amount of input data (image sequences), we coupled the described task partitioning strategy with a data partitioning strategy. Data partitioning consists in subdividing the input image sequence into disjoint subsets of images, and assigning each subset to a couple of processes devoted to the detection/removal tasks. Let $I_1,\ldots,I_K$ the sequence of K input images, each of spatial resolution $M\times N$ with a scratch of width w, and P the number (even) of available processes. The input image sequence is partitioned into $Q=P/2$ disjoint subsets $A_1,\ldots,A_Q$, each one consisting of K/Q images:

$$A_j=\{I_j,I_{j+Q},\ldots,I_{j+(K/Q-1)Q}\}=\{I_{j+iQ}\}_{i=0,\ldots,K/Q-1},\quad j=1,\ldots,Q$$

Subset A_j is assigned to the two adjacent processes P_{2j-2} and $P_{2j-1}, j=1,\ldots,Q$, and for each image belonging to A_j process P_{2j-2} performs the detection task and sends the result to process P_{2j-1}, while process P_{2j-1} receives the scratch mask from process P_{2j-2} and performs the removal task.

Communication operations among processes, which are all one-to-one, have been implemented through blocking sends and non-blocking receives, in order to overlap sends and receives and to hide latency times. Moreover, *ad hoc* high level communication routines for sending/receiving arrays of length $w\times N$, with $w\ll M$, containing exclusively the rectangular region of the scratch mask including interesting pixels have been implemented in order to reduce communication times.

The performance model of P-LSR, based on the assumption that the computational environment is homogeneous in terms of computation nodes and interconnection network, has been devised as follows. Let t_{comp} the execution time of one operation on one processor, t_{comm} the time for communication of one item between adjacent processors, and t_{lat} the latency time related to a communication between adjacent processors. The execution time of a processor can be modeled as a function of the computational complexity of its task and of the time needed for communication. An estimate of communication times can be obtained counting the number of sent/received data for each processor and adding latency times related to one communication (latency times of subsequent sends are hidden by non-blocking receives). Therefore, execution time $T_j(K,P)$ of the j-th processor for the solution to a problem of size K using P processors is given by

$$T_j(K,P)=\left[\operatorname{mod}(j+1,2)\times\frac{K}{Q}\times T_D+\operatorname{mod}(j,2)\times\frac{K}{Q}\times T_R\right]t_{comp} + \left[\frac{K}{Q}\times w\times N\right]t_{comm}+t_{lat} \tag{4}$$

where the computational complexities T_D and T_R of the detection and removal tasks are given by the sum of the computational complexities of their underlying algorithms. The total execution time $T(K,P)$ for the solution to a problem of size K using P processors is thus given by

$$T(K,P)=\max_{j=0,\ldots,P-1} T_j(K,P) \tag{5}$$

We explicitly observe that the performance model described in Eqs. (4)–(5) is sufficiently general, being related to the proposed methodology and parallelization strategy, and not to the number of adopted detection and removal algorithms. Therefore, it can be applied whichever is the number and the type of restoration algorithms taken into account for the fusion-based algorithm.

4. Experimental results

4.1. Test data

Detection and removal algorithms presented here have been tested on several real images affected by real line scratches; examples are given in Fig. 3(a), belonging to the sequence *Knight* available in [23], and in column (a) of Fig. 4, where original scratched images belong to a sequence available in [24] (first row), to the Italian movie *Animali che attraversano la strada* —2000 (second row), and to the sequence *Sitdown* available in [23] (third row). Other images affected by real line scratches together with all results obtained with the proposed approach have been made publicly available at http://www.na.icar.cnr.it/~maddalena.l/DFRLab/DataFusionScratches.html.

The considered algorithms have been tested also on artificially corrupted images in order to have known scratch masks and known ideal uncorrupted images to compare with computed detection and removal results, respectively. Specifically we considered $J=20$ sequences of $K=120$ uncorrupted original B/W images $I^j_k, j=1,\ldots,J; k=1,\ldots,K$, each of size $M_j\times N_j$, and the corresponding images with an artificial scratch of odd width w, denoted as $I^{j,w}_k, j=1,\ldots,J; k=1,\ldots,K; w=3,5,\ldots,19$, obtained as

$$I^{j,w}_k(x,y)=\begin{cases}255 & \text{if } (x,y)\in\Omega_{j,w}-\partial\Omega_{j,w}\\ 200 & \text{if } (x,y)\in\partial\Omega_{j,w}\\ I^j_k(x,y) & \text{otherwise}\end{cases} \tag{6}$$

where $\Omega_{j,w}$ denotes the scratch domain, that is the rectangular subset of the image domain of size $w\times N_j$ having as leftmost column the center column $M_j/2$ of the image: $\Omega_{j,w}=\{(x,y):x=M_j/2,\ldots,M_j/2+w-1; y=0,\ldots,N_j-1\}$, and $\partial\Omega_{j,w}$ denotes its border. Eq. (6) provides a suitable approximation of the generic line scratch [25]. One example of such images with artificial scratch of width $w=7$ is reported in Fig. 5(a), obtained from an image belonging to the sequence *Frank* available in [23]. All the other artificially scratched images together with all results obtained with the proposed approach can be found at http://www.na.icar.cnr.it/~maddalena.l/DFRLab/DataFusionScratches.html.

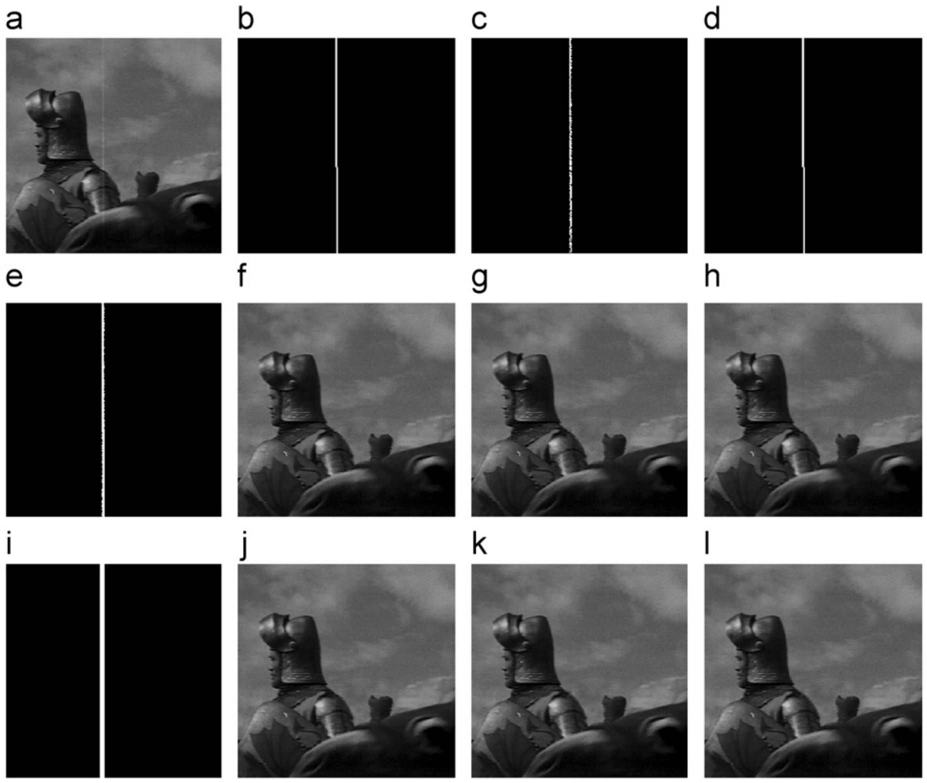


Fig. 3. Restoration of a real scratch: (a) original image with scratch of width $w=5$; scratch masks obtained by adopting (b) detection Algorithm 1, (c) detection Algorithm 2 and (d) detection Algorithm 3; (e) union fused scratch mask; restored images obtained by adopting (f) removal Algorithm 1, (g) removal Algorithm 2 and (h) LSR removal algorithm, using the union fused scratch mask; (i) MC fused scratch mask; restored images obtained by adopting (j) removal Algorithm 1, (k) removal Algorithm 2 and (l) LSR removal algorithm, using the MC fused scratch mask.

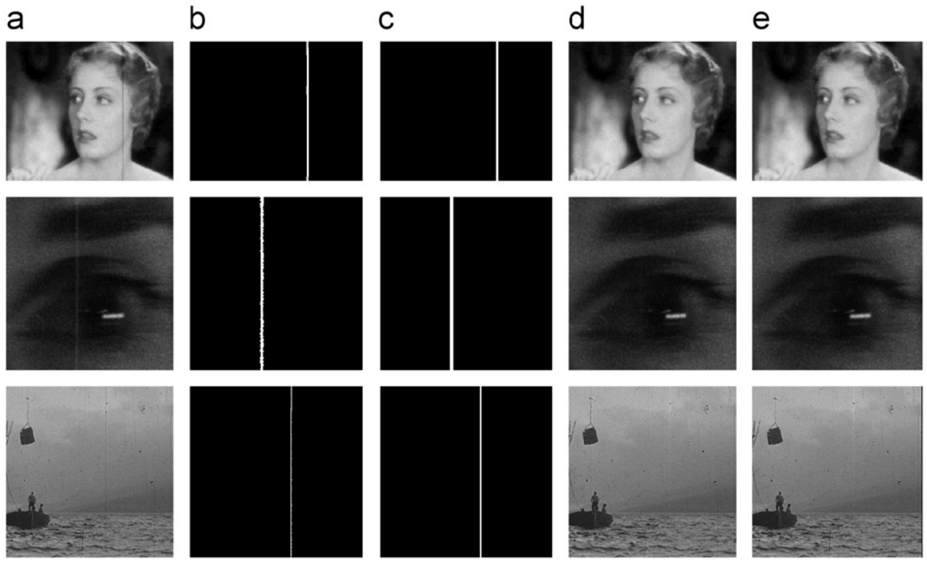


Fig. 4. Restoration of real scratches of width $w=3$ (first row), $w=5$ (second row) and $w=7$ (third row). (a) Original images; scratch masks obtained adopting (b) the union and (c) the MC detection fusion strategies; restored images obtained by adopting LSR removal algorithm using (d) the union and (e) the MC fused scratch mask.

4.2. Accuracy results

4.2.1. Detection results

The accuracy of the result of all the detection algorithms considered in Section 3.1 is quite high, as it is shown by the scratch masks reported in Fig. 3(b)–(d) for the sequence frame with real scratch of Fig. 3(a). Nonetheless, the aggregated masks reported in Fig. 3(e)and (i) seem more appropriate for the successive removal phase. Comparing the two aggregated masks, the union aggregation operator (see Fig. 3(e)) seems more appropriate for the successive removal

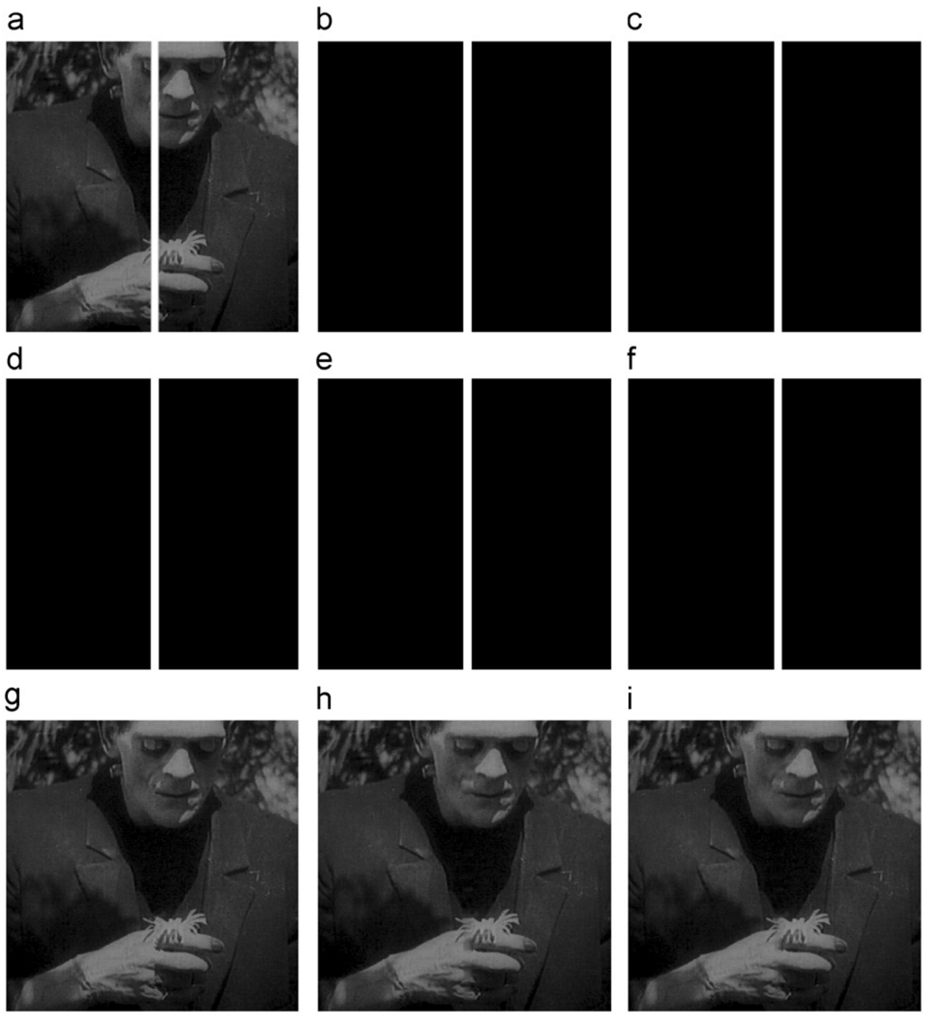


Fig. 5. Restoration of an artificial scratch: (a) image $I_1^{1,7}$ with artificial scratch of width $w = 7$ obtained as in Eq. (6); scratch masks obtained by adopting (b) detection Algorithm 1, (c) detection Algorithm 2, (d) detection Algorithm 3, (e) union and (f) MC detection fusion strategies; restored images obtained by adopting (g) removal Algorithm 1, (h) removal Algorithm 2 and (i) LSR removal algorithm, using the real scratch mask.

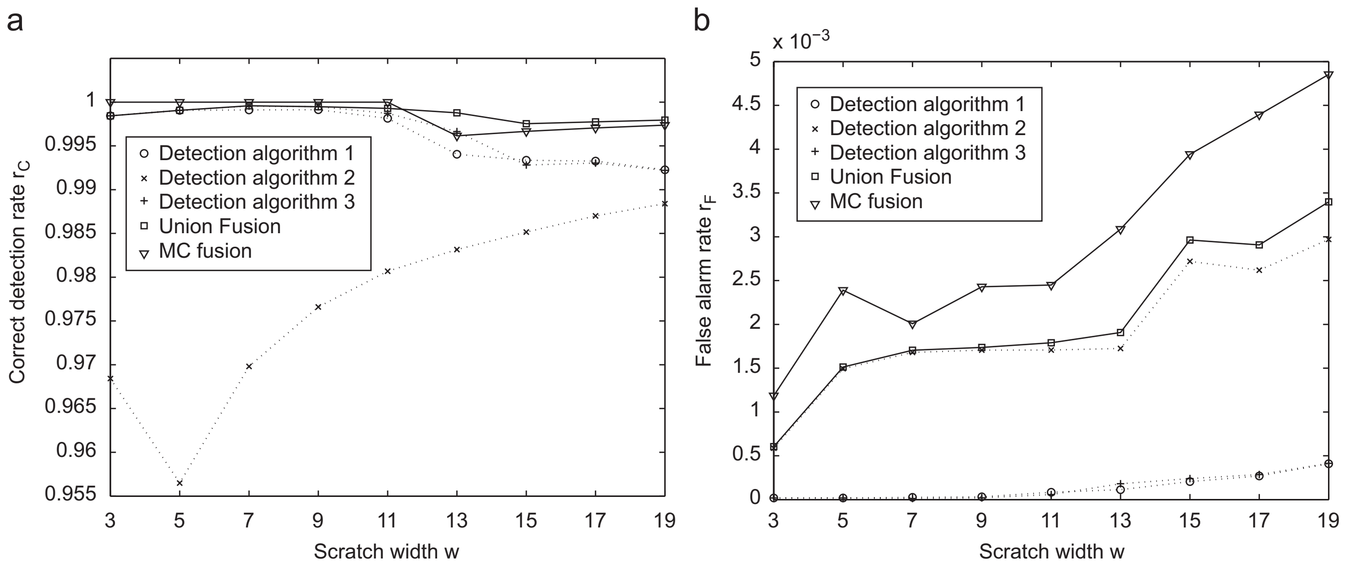


Fig. 6. Accuracy measures of the detection algorithms applied to images described in Eq. (6): (a) correct detection rates r_C and (b) false alarm rates r_F.

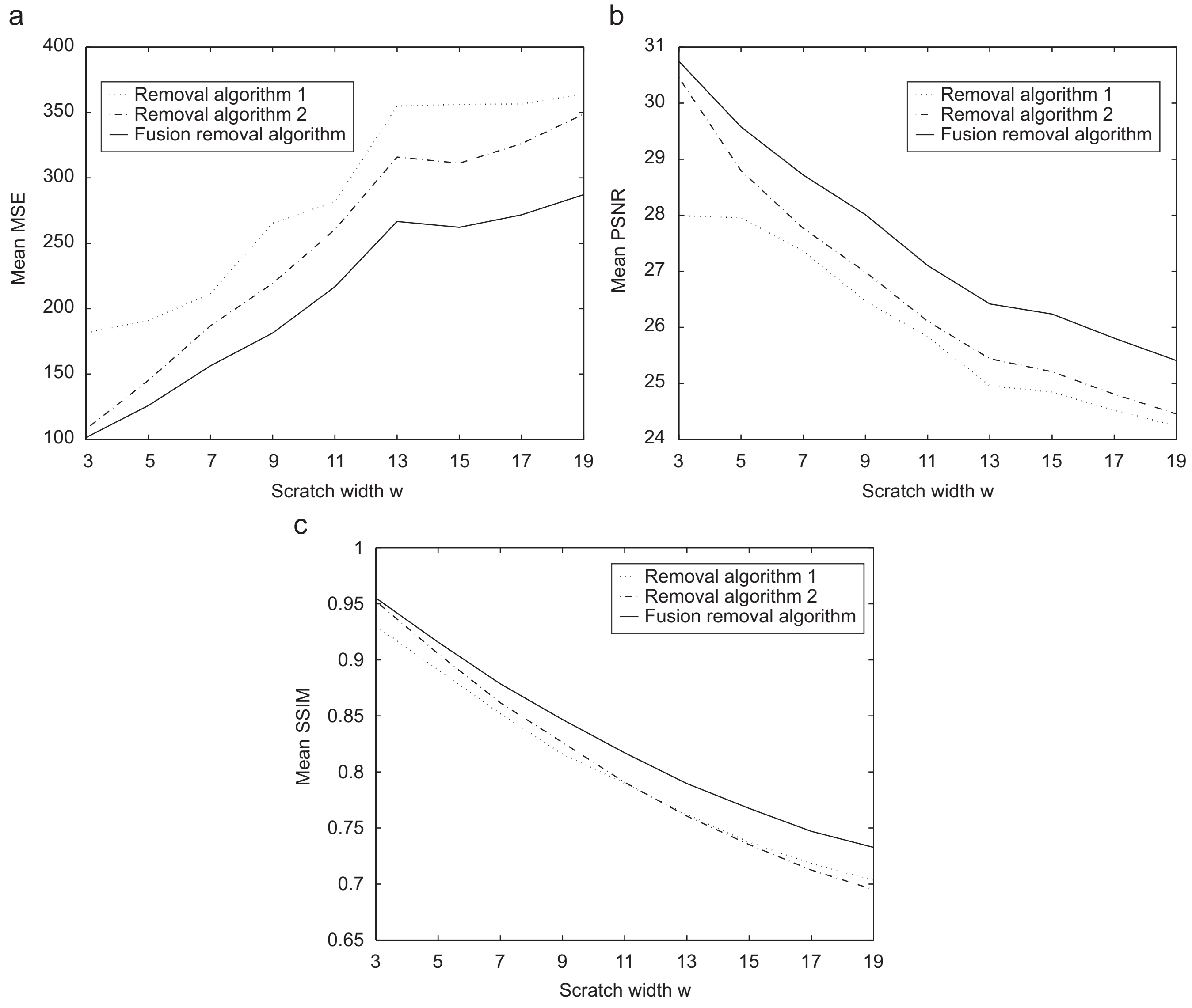


Fig. 7. Accuracy measures of the removal algorithms applied to images described in Eq. (6): (a) *MeanMSE*, (b) *MeanPSNR* and (c) *MeanSSIM*.

phase, compared with the MC aggregation operator (see Fig. 3(i)). Specifically, the union aggregation operator allows to consider all details captured by the different detection algorithms, while retaining the minimum support of the mask. The MC aggregation operator, instead, leads to a mask that appears unnatural for real images (being perfectly rectangular); moreover, if one of the underlying algorithms fails to detect the scratch, so does the aggregated mask.

Analogous considerations can be drawn from detection results for the other originally corrupted images reported in Fig. 4 and from those obtained for the artificially scratched sequence frame of Fig. 5.

In order to obtain an objective estimate of detection algorithms, we compare their results on the artificially scratched images obtained as in Eq. (6) with the real (known) scratch masks. Specifically, for each mask $B_{j,k}$ computed with anyone of the described detection algorithms for the artificially scratched image $I_k^{j,w}$ obtained as in Eq. (6) we count:

- $C_k^{j,w} = card\{(x,y) : (x,y) \in \Omega_{j,w}, B_{j,k}(x,y) = 1\}$, number of *correct detections* (pixels of the scratch that are included in the computed scratch mask);

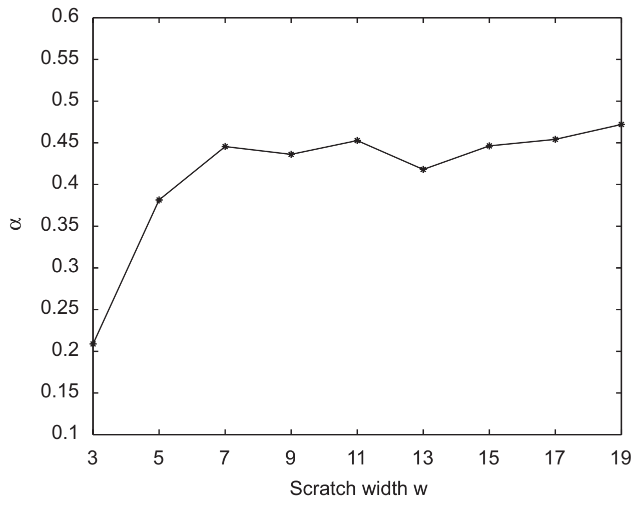


Fig. 8. Mean values of α weights in Eq. (2) which minimize the *MeanMSE* measure for the fusion removal algorithm applied to images described in Eq. (6).

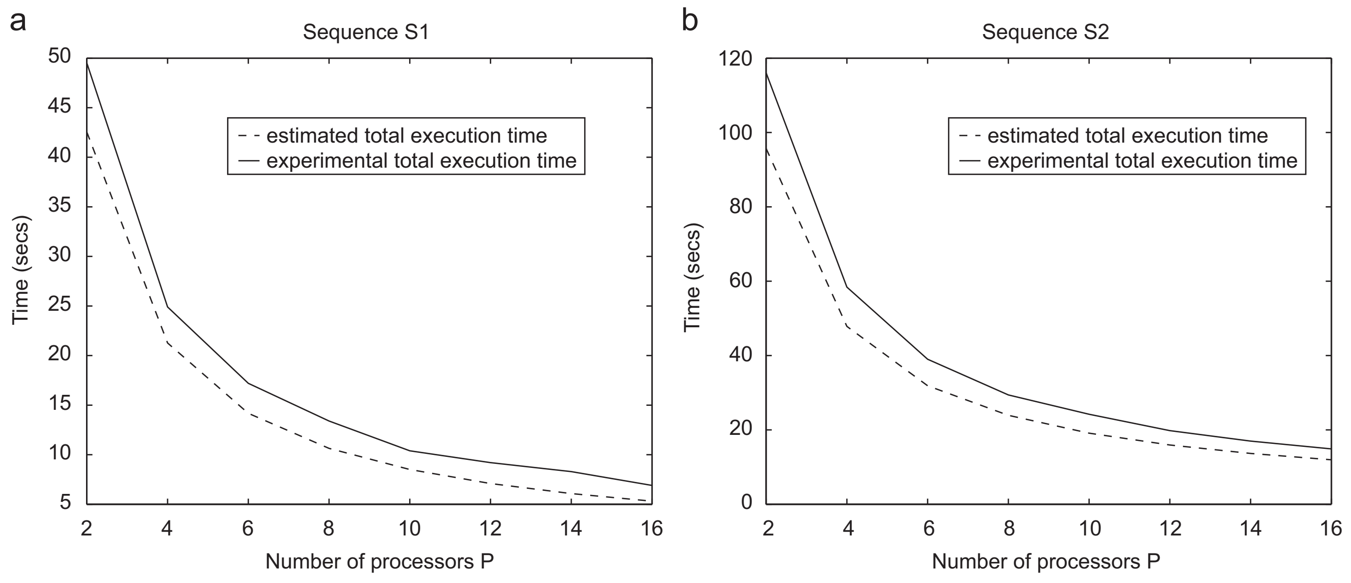


Fig. 9. Comparison of estimated and experimental execution times (s) of P-LSR for image sequences: (a) S_1 and (b) S_2.

- $F_k^{j,w} = card\{(x,y) : (x,y) \notin \Omega_{j,w}, B_{j,k}(x,y) = 1\}$, number of *false alarms* (pixels not belonging to the scratch that are included in the computed scratch mask),

and consider their rates $r_{C_k^{j,w}}$ and $r_{F_k^{j,w}}$ with respect to the reference area:

$$r_{C_k^{j,w}} = \frac{C_k^{j,w}}{w \times N_j}, \quad r_{F_k^{j,w}} = \frac{F_k^{j,w}}{M_j \times N_j - w \times N_j}$$

where $w \times N_j$ is the number of corrupted pixels (i.e. the dimension of the scratch domain $\Omega_{j,w}$), and $M_j \times N_j - w \times N_j$ is the number of uncorrupted pixels.

Given the scratch width w, the measures adopted for the objective estimation of the detection algorithms are the mean values r_C and r_F of rates $r_{C_k^{j,w}}$ and $r_{F_k^{j,w}}$ over the $J * K$ images described in Eq. (6):

- *Correct detection rate*

$$r_C = \frac{1}{J * K} \sum_{j=1}^{J} \sum_{k=1}^{K} r_{C_k^{j,w}}$$

Such measure gives values in $[0,1]$; the higher the value of r_C, the better the detection result.

- *False alarm rate*

$$r_F = \frac{1}{J * K} \sum_{j=1}^{J} \sum_{k=1}^{K} r_{F_k^{j,w}}$$

Such measure gives values in $[0,1]$; the lower the value of r_F, the better the detection result.

Values for r_C and r_F obtained with all the described detection algorithms are reported in Fig. 6, varying the scratch width w. Here we can observe that r_C values are generally very close to 1 for all detection algorithms and that only few false alarms are generated. Specifically, we observe that the union fusion strategy reaches the best r_C values achieved by the underlying detection algorithms, but worse r_F values; the MC fusion strategy reaches the best r_C values attainable ($r_C = 1$), but r_F values worse than any other algorithm.

4.2.2. *Removal results*

The results of the removal algorithms for the sequence frame with real scratch of Fig. 3(a) are reported in Fig. 3(f) and (g) for the inpainting and interpolative algorithms, respectively, based on the mask produced by the union aggregation operator; Fig. 3(j) and (k) show the results over the mask produced by the MC aggregation operator. Fig. 3(h) and (l) report the results obtained by adopting the GEM method (with $\alpha_1 = 0.3$ and $\alpha_2 = 0.7$ experimentally chosen) over the images depicted in Fig. 3(f)–(g) and (j)–(k), respectively. From these results we can observe that, even though the removal algorithms taken into account perform quite well, their reconstruction accuracy can be enhanced; the aggregated results, instead, tend to smooth the inaccuracies, still retaining the good performance of the considered algorithms. Analogous considerations can be drawn from results of the removal algorithms for the other images affected by real scratches of Fig. 4 and also for the artificially scratched sequence frame of Fig. 5, where the scratch mask adopted is the real scratch mask and the weights for the GEM aggregation operator, computed as in Eq. (3), are $\alpha_1 = 0.39$ and $\alpha_2 = 0.61$.

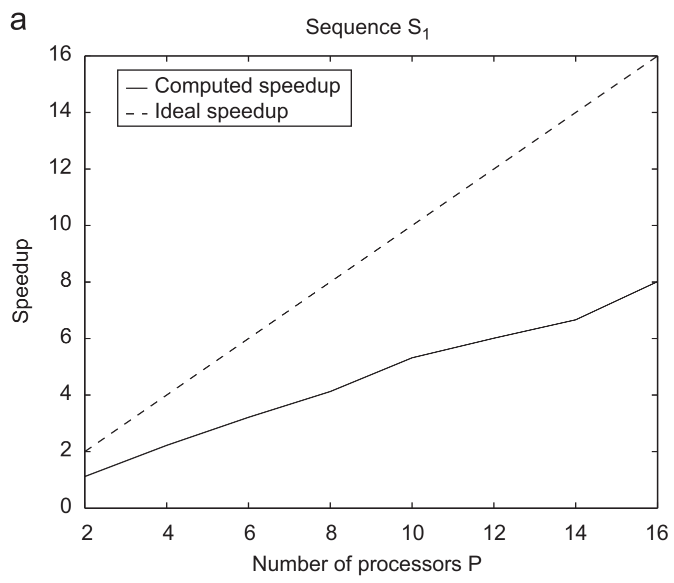
In order to obtain objective measures of the removal algorithms accuracy, we compare their results on the artificially scratched images $I_k^{j,w}$ obtained as in Eq. (6) with the ideal uncorrupted images, adopting the real (known) scratch masks. Specifically, given the scratch width w, let be, for $j = 1, \ldots, J$ and $k = 1, \ldots, K$:

- $\mathbf{o}_k^j$ the subimage of original image I_k^j containing only pixels in the scratch domain $\Omega_{j,w}$,
- $\mathbf{r}_k^j$ the subimage of the restored image R_k^j obtained with anyone of the considered removal algorithms, containing only pixels in $\Omega_{j,w}$.

In the case of LSR removal algorithm, the restored images R_k^j have been obtained choosing the GEM weights α_1 and α_2 as explained in Section 3.2.3.

Given all such images, we consider the following objective measures:

- $$MeanMSE = \frac{1}{J * K} \sum_{j=1}^{J} \sum_{k=1}^{K} \frac{1}{w * N_j} \|\mathbf{o}_k^j - \mathbf{r}_k^j\|^2$$



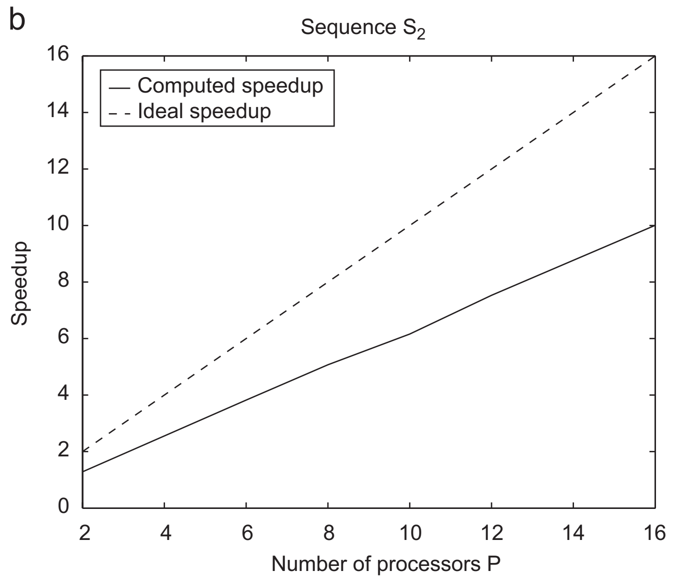


Fig. 10. Comparison of ideal and experimental speedup of P-LSR for image sequences: (a) S_1 and (b) S_2.

mean, over the $J*K$ restored images, of MSE between the original and the restored images ($\|\cdot\|$ is intended as vector norm). Such measure gives a non-negative value; the smaller the value of *MeanMSE*, the better the restoration result;

- $$MeanPSNR = \frac{1}{J*K}\sum_{j=1}^{J}\sum_{k=1}^{K}\left(10*\log_{10}\left(\frac{255^2}{\frac{1}{w*N_j}\|\mathbf{o}_k^j-\mathbf{r}_k^j\|^2}\right)\right)$$

mean, over the $J*K$ restored images, of the peak-signal-to-noise-ratio between the original and the restored images obtained considering the MSE. Such measure gives a non-negative value; the higher the value of *MeanPSNR*, the better the restoration result;

- $$MeanSSIM = \frac{1}{J*K}\sum_{j=1}^{J}\sum_{k=1}^{K}\frac{(2*\mu_{\mathbf{o}_k^j}*\mu_{\mathbf{r}_k^j}+C_1)(2*\sigma_{\mathbf{o}_k^j\mathbf{r}_k^j}+C_2)}{(\mu^2_{\mathbf{o}_k^j}+\mu^2_{\mathbf{r}_k^j}+C_1)(\sigma^2_{\mathbf{o}_k^j}+\sigma^2_{\mathbf{r}_k^j}+C_2)}$$

mean, over the $J*K$ restored images, of the structural similarity index [26] applied to the original and the restored images, where $C_1=(K_1*L)^2$, $C_2=(K_2*L)^2$, $K_1=0.01$, $K_2=0.03$ and $L=255$. Such measure gives values in $[0,1]$; the higher the value of *MeanSSIM*, the better the restoration result.

Results in terms of the described measures obtained with all removal algorithms varying the scratch width w are reported in Fig. 7. It can be observed that *MeanMSE* values obtained with the fusion removal algorithm are always lower than those obtained with the two removal algorithms and that *MeanPSNR* and *MeanSSIM* values obtained with the fusion removal algorithm are always higher than those obtained with the other two removal algorithms. Moreover, for each removal method, results obtained with all the considered measures show lower accuracy increasing the scratch width, in accordance with the increasing reconstruction difficulty as the reconstruction area widens.

In summary, we can state that the considered measures indicate the fusion method as the most accurate among the considered removal methods. Moreover, since we have considered the ideal scratch area and no one detection algorithm can produce a result better than this, we can conclude that the whole LSR restoration algorithm performs better than any combination of the considered detection and removal algorithms.

It should be stressed that, in order to fruitfully apply GEM for combining removal results, suitable values for weight α in Eq. (2) must be chosen. We could devise values for such parameter which minimize the MSE with respect to the target function (as in Section 3.2.3) since for artificially corrupted images the target function $I^*(x,y)$ (the original uncorrupted image) is known. In a generic scratch removal problem the target function is clearly not available, and care must be taken in order to choose the α weights. Therefore, we analyzed the computed values of α which minimize the *MeanMSE* measure for the fusion removal algorithm. The mean values of α over all the considered restored images varying the scratch width w are reported in Fig. 8. Such results suggest that for narrow scratches the contribution of the interpolating method to the aggregated result should be more relevant than that of the inpainting method (should be $\alpha \leqslant 0.5$), while for wider scratches both the methods should equally contribute to the final result (should be $\alpha \simeq 0.5$). In any case, no one of the two removal algorithms alone achieves better results than the fusion removal algorithm (optimal values for *MeanMSE* are never achieved for $\alpha = 0$ or 1).

4.3. Parallel performance results

For the performance evaluation of P-LSR we used a Beowulf system consisting of 20 PC's connected through a Fast Ethernet switch; each node is a Pentium IV, 1.5GHz, 512MB RAM memory and 40GB hard disk. The algorithm has been implemented in C programming language, using MPI (version 1.2.5) communication routines [27]. For validating the performance model described in Eqs. (4)–(5) we compared experimental execution times with corresponding estimates, where:

- experimental execution times have been computed as the maximum of execution times on each processor of P-LSR executed on P processors; they do not include input and output of image sequences;
- estimated execution times have been obtained from the performance model Eqs. (4)–(5) using the values:
  - $t_{comp} = 1.4\times10^{-9}$ s (time for the execution of one integer operation in the considered computational environment);

- $w \times N \times t_{comm} = 5.78 \times 10^{-4}$ s for $w = 7$ and $N = 576$, or $w \times N \times t_{comm} = 3.98 \times 10^{-4}$ s for $w = 7$ and $N = 256$ ($w \times N \times t_{comm}$ is the time for sending a message of length $w \times N$);
- $t_{lat} = 8.0 \times 10^{-5}$ s (communication latency time, computed as the time for sending a message of zero length).

Communication time t_{comm} and latency time t_{lat} related to one communication between two adjacent processors have been computed using `mpptest` program, which measures the performance of some basic MPI routines [28].

In Fig. 9 we report estimated and experimental execution times of P-LSR, varying the number of processors P, for two of the artificially scratched image sequences described in Eq. (6) and reported in Fig. 3(b) and (c): $S_1 = \{I_k^{1,7}\}_{k=1,\dots,120}$ (images of size $M \times N = 256 \times 256$) and $S_2 = \{I_k^{2,7}\}_{k=1,\dots,120}$ (images of size $M \times N = 720 \times 576$). Here we can observe that the proposed performance model is quite accurate. Moreover, in Fig. 10 we report ideal and experimental speedups of P-LSR for image sequences S_1 and S_2, varying the number of processors P. The figure highlights that acceptable speedup is achieved by P-LSR, and it gets better when the dimension of the problem grows.

5. Conclusions

This paper described a new methodology for the solution to classes of problems to be dealt with in digital movie restoration, which takes into account already existing promising algorithms and combines the obtained results through data fusion techniques in order to provide a restored sequence as similar as possible to the original uncorrupted sequence.

For the specific case of line scratch restoration, we presented the LSR sequential algorithm, based on such methodology, and described it using well-settled restoration modules and devising suitable image fusion techniques.

The proposed LSR algorithm naturally adapts for implementation into high-performance computing environments. Indeed, we described P-LSR, a parallel version of LSR algorithm based on a combination of task and data partitioning strategies, and defined a theoretical performance model of the algorithm.

The implementations of LSR and P-LSR algorithms have been tested on several corrupted and artificially corrupted real image sequences, in order to analyze the accuracy of restoration results, the performance model and the attainable speedup. The analysis has lead to the conclusion that the LSR algorithm outperforms the underlying restoration methods in terms of accuracy, the performance model for P-LSR algorithm is quite faithful, and acceptable speedups can be achieved using P-LSR, allowing for real-time restoration.

References

[1] F. Roli, J. Kittler, T. Windeatt (Eds.), Multiple classifier systems, in: Lecture Notes in Computer Science, vol. 3077, Springer, Berlin, 2004.
[2] I. Bloch, Information combination operators for data fusion: a comparative review with classification, IEEE Transactions on Systems, Man, Cybernetics Part A 26 (1) (1996) 52–67.
[3] T.K. Ho, J.J. Hull, S.N. Srihari, Decision combination in multiple classifier systems, IEEE Transactions on Pattern Analysis and Machine Intelligence 18 (1994) 66–75.
[4] M. Ceccarelli, A. Petrosino, Multifeature adaptive classifiers for SAR image segmentation, Neurocomputing 14 (1997) 345–363.
[5] A.A. Goshtasby, S. Nikolov, Special issue on image fusion: advances in the state of the art, Information Fusion 8 (2007) 113–222.
[6] S. Tilie, I. Bloch, L. Laborelli, Fusion of complementary detectors for improving blotch detection in digitized films, Pattern Recognition Letters 28 (2007) 1735–1746.
[7] A.C. Kokaram, R. Morris, W. Fitzgerald, P. Rayner, Detection of missing data in image sequences, IEEE Transactions on Image Processing Parts I–II (1995) 1496–1519.
[8] L. Joyeux, S. Boukir, B. Besserer, O. Buisson, Reconstruction of degraded image sequences. Application to film restoration, Image and Vision Computing 19 (2001) 503–516.
[9] O. Kao, J. Engehausen, Scratch removal in digitised film sequences, in: Proceedings of International Conference on Imaging Science Systems and Technology, CSREA Press, 2000, pp. 171–179.
[10] R.D. Morris, Image sequence restoration using Gibbs distributions, Ph.D. Thesis, University of Cambridge, 1995.
[11] L. Maddalena, Efficient methods for scratch removal in image sequences, in: Proceedings of 11th International Conference on Image Analysis and Processing, IEEE Computer Soc. Press, Silver Spring, MD, 2001, pp. 547–552.
[12] G. Sapiro, Image inpainting, SIAM News 35 (2002).
[13] J. Verdera, V. Caselles, M. Bertalmio, G. Sapiro, Inpainting surface holes, in: Proceedings of IEEE International Conference on Image Processing, 2003, pp. 14–17.
[14] A.A. Efros, W.T. Freeman, Image Quilting for texture synthesis and transfer, in: Proceedings of 28th Annual Conference on Computer Graphics and Interactive Techniques, 2001, pp. 341–346.
[15] A.C. Kokaram, Parametric texture synthesis for filling holes in pictures, in: Proceedings IEEE International Conference in Image Processing, 2002, pp. 325–328.
[16] R. Bornard, E. Lecan, L. Laborelli, J.-H. Chenot, Missing data correction in still images and image sequences, in: Proceedings of ACM Multimedia, 2002, pp. 355–361.
[17] G. Laccetti, L. Maddalena, A. Petrosino, Removing line scratches in digital image sequences by fusion techniques, in: F. Roli, et al. (Eds.), 13th International Conference on Image Analysis and Processing (ICIAP2005), Lecture Notes in Computer Science, vol. 3617, Springer, Berlin, Heidelberg, 2005, pp. 695–702.
[18] M.P. Perrone, L.N. Cooper, When networks disagree: ensemble method for neural networks, in: R.J. Mammone (Ed.), Artificial Neural Networks for Speech and Vision, Chapman & Hall, New York, 1993, pp. 126–142.
[19] G. Piella, Adaptive wavelets and their applications to image fusion and compression, Ph.D. Thesis, CWI and University of Amsterdam, 2003.
[20] P.M. de Zeeuw, G. Piella, H.J.A.M. Heijmans, A Matlab toolbox for image fusion (MATIFUS), Report PNA-E0424, CWI, Amsterdam, 2004.
[21] G. Laccetti, L. Maddalena, A. Petrosino, P-LSR: a parallel algorithm for line scratch restoration, in: Proceedings of the Seventh International Workshop on Computer Architecture for Machine Perception (CAMP2005), IEEE Computer Soc. Press, Silver Spring, MD, 2005, pp. 225–230.
[22] S.S.S. Ramaswamy, P. Banerjee, A framework for exploiting task and data parallelism on distributed memory multicomputers, IEEE Transactions on Parallel and Distributed Systems 8 (1997) 1098–1115.
[23] A.C. Kokaram, Motion Picture Restoration: Digital Algorithms for Artefacts Suppression in Archived Film and Video, Springer, Berlin, 1998.
[24] DIAMANT Project (EC IST 1999 12078)—digital film manipulation system ⟨http://diamant.joanneum.ac.at/⟩.
[25] L. Maddalena, A. Petrosino, Restoration of blue scratches in digital image sequences, Image and Vision Computing 26 (2008) 1314–1326.
[26] Z. Wang, L. Lu, A.C. Bovik, Video quality assessment based on structural distortion measurement, Signal Processing: Image Communication 19 (2004) 121–132.
[27] M. Snir, S. Otto, S. Huss-Lederman, D. Walker, J. Dongarra, MPI—The Complete Reference, vol. 1, second ed., The MPI Core, 1998.
[28] W. Gropp, E. Lusk, Reproducible measurements of MPI performance characteristics, in: Proceedings of 6th European PVM/MPI Users' Group Meeting on Recent Advances in Parallel Virtual Machine and Message Passing Interface, 1999, Lecture Notes in Computer Science, Springer, Berlin, pp. 11–18.

About the author—LUCIA MADDALENA received the Laurea degree cum laude in Mathematics and the Ph.D. in Applied Mathematics and Computer Science from the University of Naples Federico II, Naples, Italy. She is a Staff Researcher at the Institute for High-Performance Computing and Networking, National Research Council of Italy. Her research interests include image processing, multimedia systems and parallel computing.

About the author—ALFREDO PETROSINO is an Associate Professor of Computer Science, University of Naples Parthenope. His research interests include image processing, pattern recognition, neural networks and multimedia systems. He is the author of more than 70 referred papers and co-editor of five international volumes. He is a Senior Member of IEEE and Member of IAPR.

About the author—GIULIANO LACCETTI is a Full Professor of Computer Science, University of Naples "Federico II". His research interests include parallel computing, grid computing and scientific computing. He is the author of a lot referred papers in these fields and he is a Member of IEEE.