Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

Restoration of blue scratches in digital image sequences

Lucia Maddalena, Alfredo Petrosino

RT-ICAR-NA-05-21 **December 2005**

Consiglio Nazionale delle Ricerche, Istituto di Calcolo e Reti ad Alte Prestazioni (ICAR) – Sede di Napoli, Via P. Castellino 111, 80131 Napoli, URL: *www.na.icar.cnr.it*

Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

Restoration of blue scratches in digital image sequences[1]

Lucia Maddalena[2], Alfredo Petrosino[3]

Rapporto Tecnico N.:
RT-ICAR-NA-05-21

Data:
dicembre 2005

[1] Sottomesso per pubblicazione

[2] Istituto di Calcolo e Reti ad Alte Prestazioni, ICAR-CNR, Sede di Napoli, Via P. Castellino 111, 80131 Napoli

[3] Università di Napoli Parthenope, Dipartimento di Scienze Applicate, Via A. De Gasperi 5, 80133 Napoli

Restoration of blue scratches in digital image sequences[*]

LUCIA MADDALENA[1], ALFREDO PETROSINO[2]

[1]National Research Council, ICAR
Via P. Castellino 111, 80131 Naples, ITALY, Tel.: +39-081 6139522; fax: +39-081 6139531
lucia.maddalena@na.icar.cnr.it

[2]University Parthenope of Naples, Department of Applied Science
Via A. De Gasperi 5, 80133 Naples, ITALY, Tel.: +39-081 5476601; fax: +39-081 5522293
alfredo.petrosino@uniparthenope.it

Abstract

In this paper we consider the problem of detecting and removing blue scratches from digital image sequences. In particular, we propose a detection method and a removal method that strongly rely on the specific features of such scratches. Evaluation of the proposed methods, in terms of both accuracy and performance timings, and numerical experiments on real images are reported.

Keywords: Colour digital film restoration; Blue scratch; Scratch detection; Scratch removal

1. Introduction

Digital Film Restoration is an evolving area of Image Processing aimed at studying methodologies and techniques that allow to digitally restore damaged movies, in order to preserve their historical, artistic and cultural value and to facilitate their diffusion through modern communication media.

Several types of defects can be found in a damaged movie, such as dust and dirt, brightness and positional instability, colour fading, scratches. We are specifically concerned with persistent *scratches*, intended as vertical lines appearing at the same location in subsequent frames of the image sequence. White or black scratches in old movies are mainly due to the abrasion of the film caused by spurious particles present in the camera, during the sequence acquisition phase, or in the projector, during the film projection. Instead, blue scratches, which are the subject of our interest, affect many modern colour movies and are due to spurious particles present in the transport mechanism of the equipment used for the development of the film.

Most of the methods reported in literature that afford this kind of problem are articulated in a *detection* phase and a *removal* phase.

[*] Submitted for publication

The detection phase consists in searching, among all the vertical lines of the images, those that are not natural lines of the scene, which are characterized as defects. Several methods have been adopted in the case of white or black scratches, such as those based on low/high pass filters [17, 18], morphological filters [9, 12-15, 27], adaptive binarization [16], discrete wavelet decomposition [5], statistics and MAP techniques [24, 25, 30], or local gradient measures in the image [1, 21] or in the image cross section [6], eventually coupled with techniques such as Hough transform [15, 18, 21] or Kalman filter [12-15], and possibly followed by Bayesian refinement strategies [18]. The result of the detection phase over a sequence frame is a binary image, the *scratch mask*, of the same size, where white pixels are related to scratch pixels in the corresponding sequence frame.

The removal phase consists in reconstructing corrupted information in the defect area individuated by the scratch mask. Depending on the amount of the defect, information included in the scratch area can be either slightly or strongly affected by the defect; thus, the scratch removal problem can be approached either as a *partially corrupted data* problem or as a *missing data* problem, respectively. Following the *partially corrupted data* approach, information included in the artifact area is taken into account for the removal. In the case of black or white scratches, some authors adopted such approach and obtained removal through morphological filters [27], interpolation or approximation [14, 15, 24], eventually followed by the reconstruction of high-frequency components via Fourier series [14] or via MAP techniques [15]. On the other hand, in the *missing data* approach pixels in the artifact area are considered missing even if they are only slightly altered. This approach has been adopted for black or white scratches by many authors, who obtained removal through interpolation or approximation [5, 22, 26], the adoption of autoregressive models [17, 18], morphological filters [9], or mean vector filters [16], eventually with the addition of least squares-based grain estimation [22]. Moreover, this approach is the one generally adopted for image *inpainting*, that is the set of techniques for making undetectable modifications to images [28]; such techniques are generally used to fill-in missing data or to substitute information contained in small image regions [32]. Inpainting has been pursued in literature also under different names, such as *image interpolation* (e.g. [29]) and *fill-in* (e.g. [2, 19]); the problem has been afforded also as *disocclusion*, since missing data can be considered as occlusions hiding the image region to be reconstructed (e.g. [3, 23]). Finally, 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 [11]; specifically, inpainting can be considered as a constrained texture synthesis problem [4, 19, 20].

Even though the problem of detection and removal of white or black scratches in digital image sequences has been considered by so many authors and several commercial software systems include modules for their restoration (such as the *DIAMANT Suite* distributed by HS-ART Digital Service GmbH or the *Revival* distributed by da Vinci Systems, Inc.), the specific case of blue scratches has not been specifically addressed. As already mentioned, they generally affect modern colour movies and, therefore, before launching a new motion picture, the film must be digitally restored by companies specialized in digital effects and post-processing. The need for efficient and automatic tools able to digitally remove blue scratches has been the primary input for the reported research. Specifically, in this paper we propose a method for the detection and removal of blue scratches in digital images that takes into account the specific features of such kind of scratches.

The contents of this paper are as follows. In Section 2 the features of blue scratches are analysed, in order to device suitable digital restoration techniques. Sections 3 and 4 outline the methods that we propose for blue scratch detection and removal, respectively, giving details of the related algorithms and implementations. In Section 5 we describe qualitative and quantitative results achieved by the proposed approach on real images. Conclusions are reported in Section 6.

2. Blue scratch characterisation

Blue scratches in a digital image sequence appear as blue strips located along a thin area covering from top to bottom of each sequence frame. Examples of blue scratches are given in Figs. 1, 3 and 4, which are details of 24 bits RGB colour images, originally of size 2880x2048, belonging to the movie *Animali che attraversano la strada* (2000).

Contrary to white or black scratches appearing in dated movies, the direction of blue scratches does not deviate too much from the vertical direction, and their position along the horizontal direction does not change too much (no more than few pixels) from one frame to the next. Therefore usually blue scratches are not oblique and have fixed position in consecutive frames of the image sequence. This is due to the fact that blue scratches are not caused by improper storage conditions or improper handling of the film, as is usually the case for ancient movies. They are rather caused by spurious particles present in the transport mechanism of the development equipment; in the case of modern equipment, the transport mechanism strictly controls the slippage of the film, which cannot move too much from its rectilinear trajectory. Due to this feature, restoration of blue scratches cannot rely on temporal discontinuity of the image intensity function along the sequence; therefore, in the following we concentrate on purely spatial scratch detection and removal in each image.

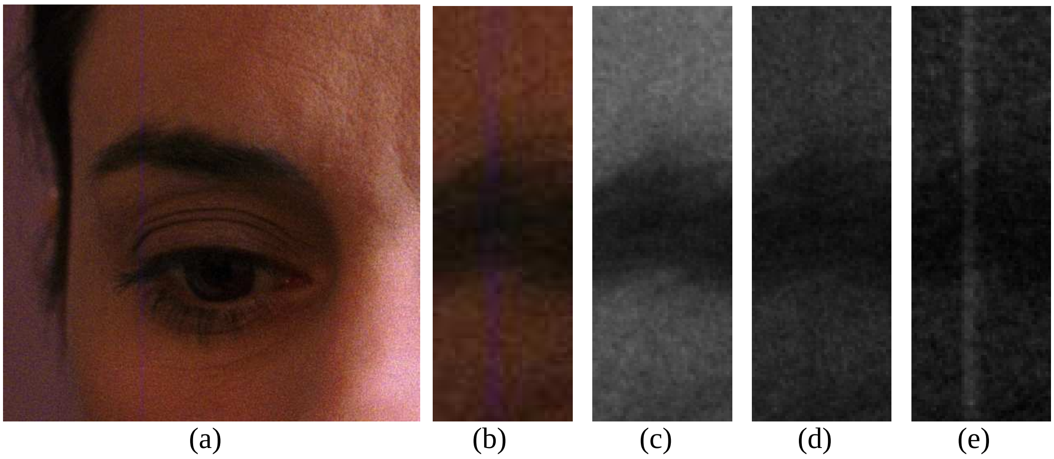
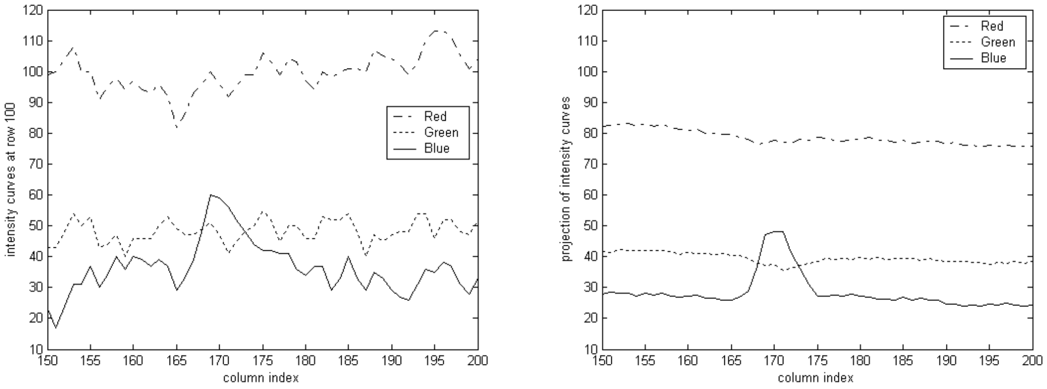


Fig. 1: Example of a blue scratch: (a) colour image; (b) scratch detail; (c) red, (d) green, and (d) blue band of scratch detail.

Inside the blue scratch area, original information has been substituted by more or less intense blue colour. Specifically, considering the RGB colour space, in the blue band there are increased intensity values compared with the neighbourhood of the scratch; in the green band some of the pixels are altered in an unpredictable way, usually with a slight increase or decrease of intensity values; the red band is usually uncorrupted, although sometimes there could be small fluctuations of intensity values in pixels belonging to the scratch area. A detail of a blue scratch and its red, green and blue bands is given in Fig. 1.

In order to have a better understanding of the scratch structure, we have analysed all corrupted sequences of the above mentioned movie, identifying three types of blue scratches. The most common type includes blue scratches such as the one appearing in Fig. 1. Looking at Fig. 2-(a), which shows the intensity curve of each colour band of the image of Fig. 1, taken as horizontal section of the image intensity function at row 100, it clearly appears that the intensity curve of the blue band has a ridge in the scratch area. The described effect is still more evident in Fig. 2-(b), where the horizontal projection of the intensity curve, taken as the mean over the image columns of the intensity curve, is shown for the three colour bands. Specifically, in the scratch area the projection of the blue band has a ridge whose width *w* is about 9 pixels and whose height *h* is about 25 intensity values; the projection of the green band presents a slight decrease of about 5 intensity values around the center of the scratch. The projection of the red band does not show clear effects of the scratch, and red band can be therefore considered as uncorrupted.

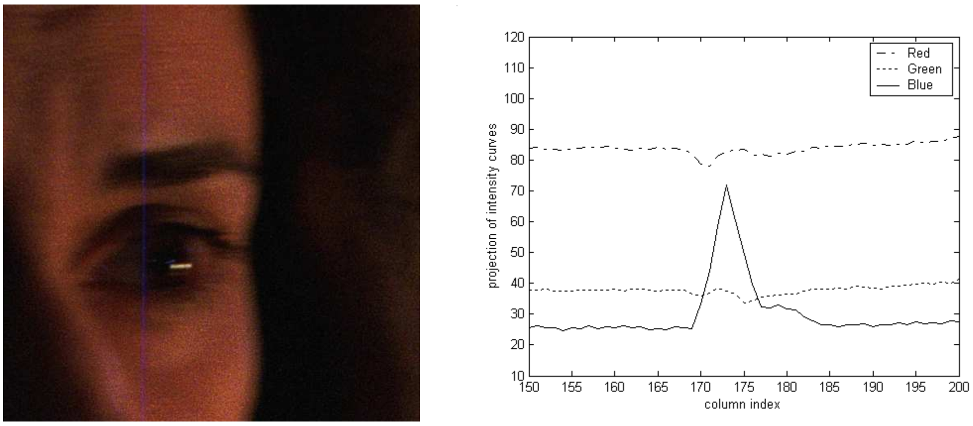


(a) (b)

Fig. 2: Profiles of the blue scratch in the image of Fig. 1: (a) intensity curves of the three bands, taken at row 100; (b) horizontal projection of the image intensity curves of the three bands.

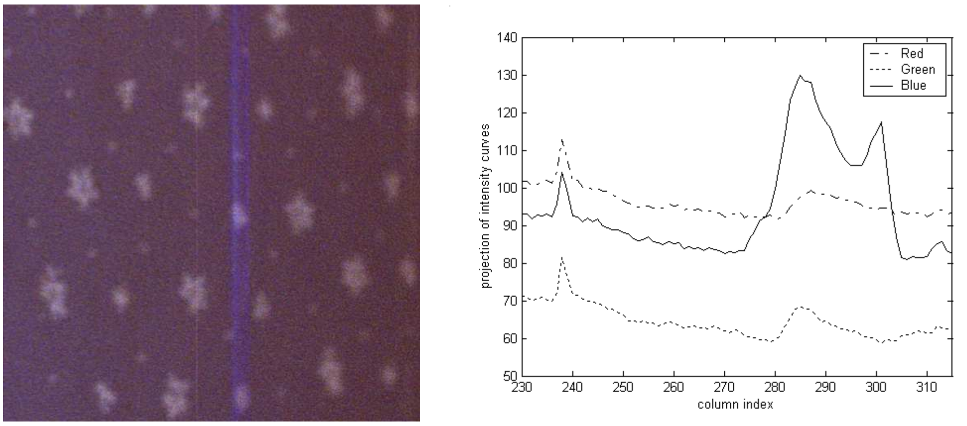
The second type includes less common blue scratches, as the one appearing in Fig. 3. Here we can observe that in the scratch area the projection of the blue band has a ridge accompanied by a “shadow” on the right; the total scratch width *w* is about 15 pixels, while the ridge height *h* is about 50 intensity values. The projections of the green and red bands show small fluctuations of about 5 intensity values in the scratch area.

The third type includes less common blue scratches that appear as two scratches close together, as the one presented in Fig. 4. Here we can observe that in the scratch area the projection of the blue band has two neighboring ridges whose cumulative width is about 29 pixels, and whose heights are about 45 and 35 intensity values respectively; the projection of the green band presents a slight increase of about 10 intensity values around the center of the left ridge. The projection of the red band does not show clear effects of the scratch, and red band can be therefore considered as uncorrupted. In Fig. 4 it is also interesting to observe that the white scratch appearing on the left of the blue one has colour band horizontal projections different from those of the blue scratch, since for white scratches the ridge affects all three colour bands in the same way.



(a) (b)

Fig.3: Example of a blue scratch: (a) colour image; (b) horizontal projection of the image intensity curves of the three bands.



(a) (b)

Fig.4: Example of a blue scratch: (a) colour image; (b) horizontal projection of the image intensity curves of the three bands.

3. Blue scratch detection

3.1. Description of the method

The idea at the basis of the blue scratch detection algorithm is that of searching, among all pixels belonging to vertical lines of the image, those having an *intense blue colour*.

Specifically, our method consists in enhancing vertical edges of the image by applying a suitable local operator, and, due to the specific features of blue scratches, in restricting the search to vertical ridge edges, whose pixels are local maxima for intensity curves of the blue band along the horizontal direction. This restriction allows to avoid considering contours of scene objects that appear as vertical lines but that are not image defects. The process leads to a modified version I_E of the original image, where blue vertical lines are particularly emphasized.

In order to discriminate between pixels belonging to eventual blue vertical lines of the scene and pixels belonging to the blue scratch, we should be able to determine the "*intense blue colour*" that is proper of blue scratches as emphasized in I_E. The HSV colour space, which relies on the hue, saturation and value properties of each colour, allows to specify colours in a way that is close to human experience of colours. Therefore, the conversion of the image to the HSV space can be helpful in finding the blue colour range.

Once the range of the blue colour searched has been determined, a binary image I_B is obtained from the enhanced image I_E, where pixels are marked if their colour is in this range.

Finally, we identify the abscissae of vertical lines of image I_B (and therefore those of vertical blue scratches of the original image) as local maxima of the horizontal projection of I_B.

Further improvement in the above described procedure can be obtained if the input image is suitably pre-processed and if the resulting scratch mask is suitably post-processed. The preprocessing is aimed at reducing noise that could affect the input image, due to film grain, dust and dirt, digitisation artifacts, etc.; the post-processing is aimed at refining the scratch mask.

3.2. BSD algorithm

Let I be the RGB image:

$$I = \{I(i, j, k), i = 1,\ldots, N; j = 1,\ldots, M; k = 1,2,3\},$$

where N is the image height, M is the image width, and $k=1, 2,$ and 3 correspond to red, green, and blue bands respectively. The proposed Blue Scratch Detection (BSD) algorithm for the detection of blue scratches in a digital image I is the following:

BSD Algorithm

Step 1. Pre-processing of the input image: noise reduction, with preservation of vertical edges;

Step 2. Enhancement of vertical blue lines:

a. enhancement of vertical edges;
b. elimination of vertical edges not produced by vertical blue lines;

Step 3. Binarisation: for each pixel of the image intensity matrix resulting from step 2:

a. convert from RGB space to HSV space;
b. if HSV values correspond to the intense blue colour, set to 1 the corresponding pixel in the binary image I_B;

Step 4. Refinement of the scratch mask: detection in the binary image I_B of vertical lines that cover almost the whole image height.

In our experiments, for Step 1 we apply a one-dimensional low-pass filter along the columns of the image intensity function, so that vertical edges are preserved. The filter adopted is the mean in a 11 pixels vertical neighbourhood of each pixel. The preprocessed image I_P resulting from Step 1 applied to the image of Fig. 1 is shown in Fig. 5-(a).

For Step 2.a we apply a one-dimensional high-pass filter along the rows of the image intensity function. Supposing that w is the scratch width in the i-th row, for each pixel $I_P(i,\ j,\ \cdot)$ the filter adopted for our experiments is the filter in a $3w$ pixels neighbourhood whose result is described as:

$$I_E(i,j,\cdot) = \sum_{l=j-3w/2}^{j+3w/2} a_l I_P(i,l,\cdot),$$

where:

$$a_l = \begin{cases} 2 & l = j - w/2,\ldots,j + w/2 \\ -1 & otherwise \end{cases}.$$

In Step 2.b we want to restrict our attention only to vertical edges produced by blue vertical lines of width w; that is, we want to consider only vertical edges whose blue band horizontal profile is a ridge edge of width w. For each pixel $I_P(i, j,\ \cdot)$ we consider the three quantities:

$$S_L(k) = \sum_{l=j-3w/2}^{j-w/2-1} I_P(i,l,k), \qquad S_C(k) = \sum_{l=j-w/2}^{j+w/2} I_P(i,l,k), \qquad S_R(k) = \sum_{l=j+w/2+1}^{j+3w/2} I_P(i,l,k),$$

and, for $k=1, 2, 3$, set $I_E(i, j, k) = 0$ if $S_C(3) < S_L(3)$ or $S_C(3) < S_R(3)$. This strong condition, in fact, ensures that the pixel $I_P(i, j,\ \cdot)$ does not belong to a vertical ridge edge of width w of the blue band of image I_P.

Note that Steps 2.a and 2.b can be merged in a single step, where for each pixel $I_P(i,\ j,\ \cdot)$ we compute the above quantities $S_L(k)$, $S_C(k)$, and $S_R(k)$ and we set:

$$I_E(i,j,k) = \begin{cases} -S_L(k) + 2 * S_C(k) - S_R(k) & if \quad S_C(3) > S_L(3) \ \ and \ \ S_C(3) > S_R(3) \\ 0 & otherwise \end{cases}.$$

The result of Step 2 on the image of Fig.1 is reported in Fig. 5-(b).

The conversion from the RGB space to the HSV space adopted in Step 3.a is computed as follows:

$$\mathrm{V} = \max(\mathrm{R},\mathrm{G},\mathrm{B}), \qquad S = \begin{cases} 0 & V = 0 \\ [V - \min(R,G,B)]/V & V \neq 0 \end{cases}, \qquad H = \begin{cases} 0 & S = 0 \\ 60(G-B)/(S*V) & V = R \\ 60[2+(B-R)]/(S*V) & V = G \\ 60[4+(R-G)]/(S*V) & V = B \\ H+360 & H < 0 \end{cases}$$

where, for each pixel, the input values R, G, B are the pixel intensity values in the three bands, normalized in [0,1], and the output values H, S, V are such that H$\in$[0,360], S$\in$[0,1], and V$\in$[0,1]. The blue colour is searched among pixels having hue H$\in$[180,300] (240 being the blue hue), saturation S > 0.45 and value V >

0.1; these values take into account the transformations performed on the original image I for obtaining the enhanced image I_E, and have been experimentally chosen performing tests on several different images. The binary image I_B resulting from Step 3 on the image of Fig. 1 is reported in Fig. 5-(c).

In Step 4 we detect vertical lines of the binary image I_B as local maxima of the horizontal projection P of I_B, whose j-th element belonging to the generic band is defined as:

$$P(j;\cdot) = \sum_{i=1}^{N} I_B(i, j;\cdot), \qquad j = 1,\ldots,M .$$

Since blue scratches usually cover most of the height of the image, a local maximum for P in column j should be obtained for $P(j, \cdot)$ close to the image height N. Therefore, we eliminate from the scratch mask the whole column j as soon as $P(j, \cdot)$ is lower than a fixed percentage of N. We experimented that, in order to avoid deleting from the mask the scratch contours, obtaining a too slim mask, it is better to fix a percentage value lower than 100% of the height. In the general case, a percentage equal to 50% is a good compromise between lack of false positives and accurate detection of the blue scratch (see for instance Fig. 5-(d)).

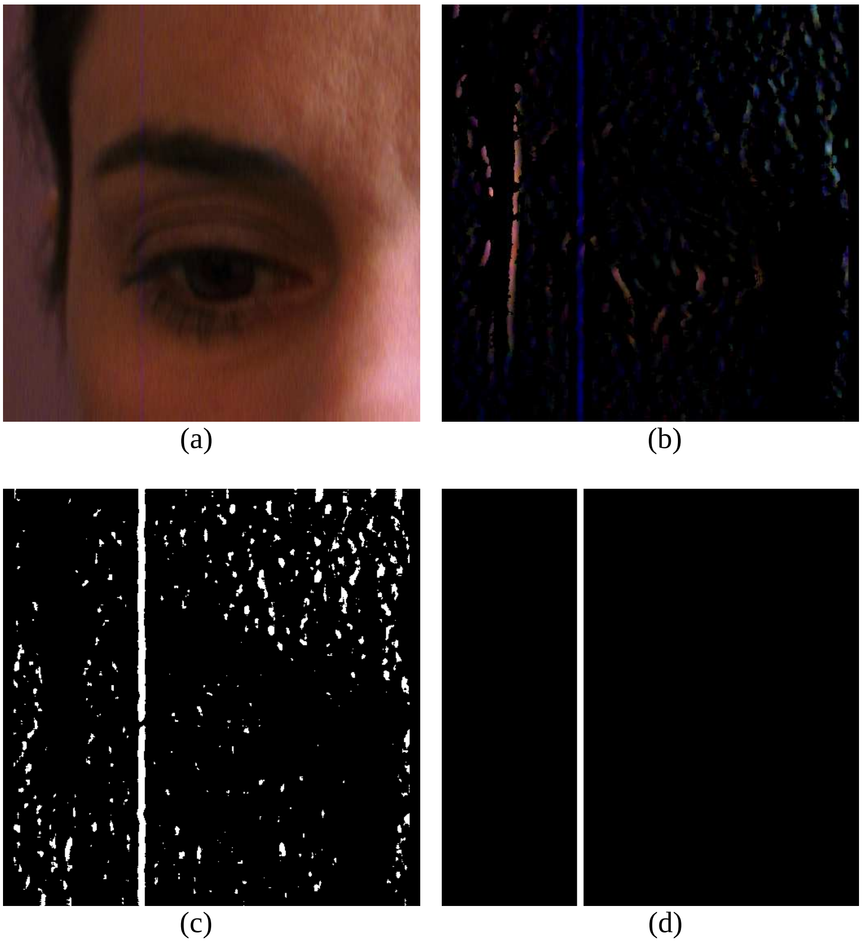

(a)
(b)
(c)
(d)

Fig. 5: BSD algorithm for the image of Fig. 1: Results of (a) Step 1; (b) Step 2; (c) Step 3; (d) Step 4.

It should be explicitly observed that, in order to have an automatic restoration algorithm, the scratch width is preliminary computed using local minima/maxima of the luminance cross-section, as in [6]. Other techniques, such as those used in [15, 18, 24, 30], could be alternatively adopted.

4. Blue scratch removal

4.1. Description of the method

In analysing the blue scratch features, we have already observed in Section 2 that pixels belonging to the scratch have undergone an intensity value reduction or increase (depending on the considered colour band) compared with pixels in the scratch neighbourhood, but still retain useful information concerning the image structure. Therefore we approach the blue scratch removal problem as a *partially corrupted data* problem.

Looking more into details at plots reported in Figs. 2, 3, and 4, we can observe that in uncorrupted areas of the image the displacements of the blue band intensity values from those of the red band are locally roughly constant; the same holds for displacements of the green band from the red band. In the scratch area, instead, such displacements appear strongly varying. Since, as already observed in Section 2, the red band is usually uncorrupted, we can restore the green and blue bands bringing their displacement from the red band inside the scratch area to the same displacement they have outside the scratch area.

4.2. BSR algorithm

The Blue Scratch Removal (BSR) algorithm we have designed can be sketched as follows:

BSR Algorithm

```
For each row of the image:
Step 1. preprocessing of the red band;
Step 2. compute minimum, maximum and median displacement of the green and blue bands from
        the red band in an uncorrupted neighbourhood of the scratch;
Step 3. add median displacement to all pixels of the green and blue bands belonging to the scratch
        area whose displacement from the red band is below minimum or above maximum
        displacement.
```

Step 1, here accomplished with rank-order filters, is required to take into account cases where the red band appears slightly corrupted.

For Step 2 of BSR algorithm in the *i*-th row the neighbourhood $N_{i,k}$ for band *k* chosen in our experiments consists of three uncorrupted pixels belonging to the same row on the right of the scratch and three on the left:

$$N_{i,k} = \{I(i,j,k):\ j = b-3, b-2, b-1, b+w, b+w+1, b+w+2\},$$

where *w* is the scratch width and *b* indicates the first column of the scratch.

Defining the displacement in the *i*-th row of the band *k* from the red band as:

$$D_{i,k} = \{d(i,j,k) = I(i,j,1) - I(i,j,k) : I(i,j,k) \in N_{i,k}\},$$

in Step 2 we compute:

$$D^{max}_{i,k} = \underset{d(i,j,k)\in D_{i,k}}{max} \{d(i,j,k)\}, D^{min}_{i,k} = \underset{d(i,j,k)\in D_{i,k}}{min} \{d(i,j,k)\}, D^{med}_{i,k} = median\{D_{i,k}\},$$

and in Step 3 we restore the *k*-th colour component *I(i, j, k)* of a pixel as:

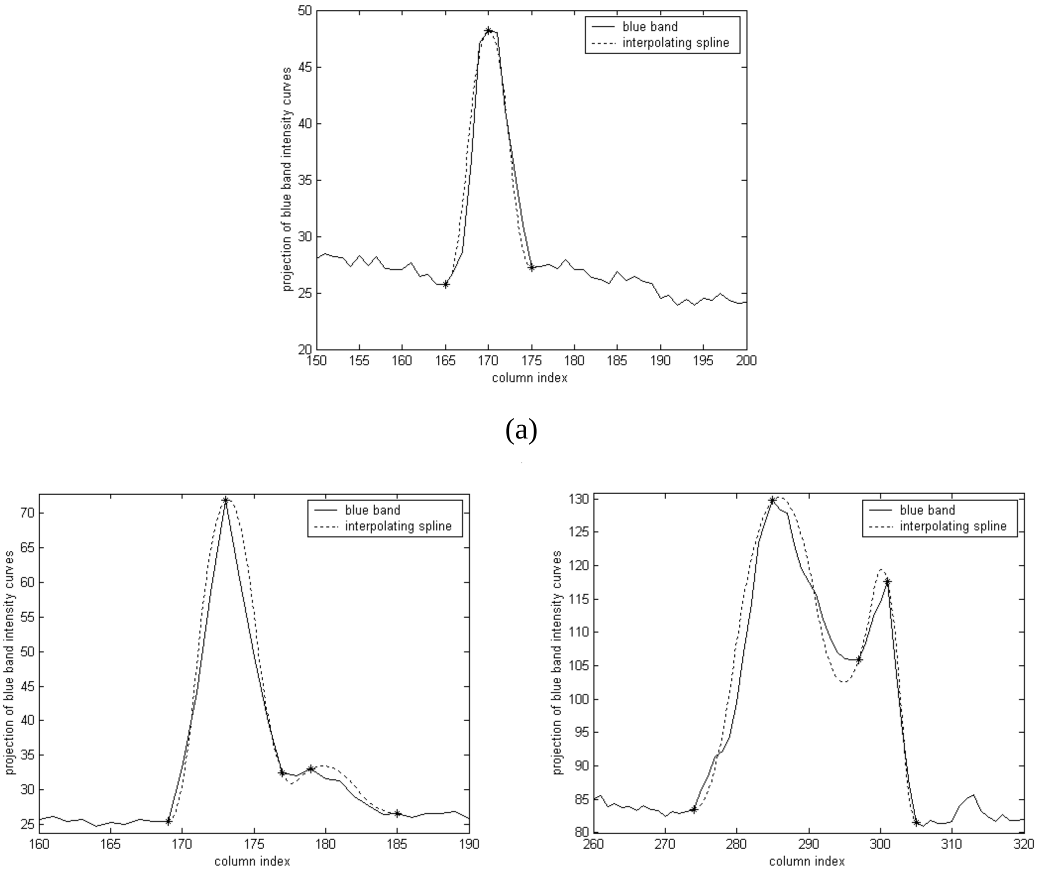
$$I(i,j,k) = I(i,j,1) - D^{med}_{i,k}$$

if its value is not included in [$D^{min}_{i,k}$, $D^{max}_{i,k}$].

5. Experimental results

5.1. Evaluation of BSD algorithm

BSD algorithm has been tested on several real images. From the visual inspection standpoint, the accuracy of the achieved results appears quite high, as it is shown by the scratch mask reported in Fig. 5-(d) for the image of Fig. 1.



(a)

(b)

(c)

Fig. 6: Complete cubic interpolating spline models for blue band horizontal projection of the images in:(a) Fig. 1; (b) Fig. 3; (c) Fig. 4.

Anyway, the validity of the method claims for a more quantitative evaluation. To this aim, we have artificially corrupted real images with blue scratches. We modelled the horizontal projection of the blue band in the scratch with a complete cubic spline interpolating extrema of the projection and its maximum point. Such model is quite adequate for the general blue scratch, as it is shown in Fig. 6-(a), where the complete cubic spline interpolating points marked as '*' is superimposed to the real blue band projection of the image in Fig. 1-(a). Different blue scratch profiles, such as those presented in Figs. 3-(b) and 4-(b), can be analogously modelled with a complete cubic spline interpolating suitable points, as it is shown in Figs. 6-(b) and 6-(c). Moreover, since the behaviour of the green band projection cannot be modelled a priori, to create more realistic artificial blue scratches for the green band projection we apply a similar model, scaled by a factor *f*.

Specifically, we considered $L=20$ uncorrupted original RGB images I_l, $l=1, \ldots, L$, each of size $N_l \times M_l$,; they include well known images (e.g. 'Lena', 'Tiffany') obtained by [7, 8, 31] as well as images taken from uncorrupted areas of already digitised images of the movie *Animali che attraversano la strada* (2000). The corresponding images with an artificial blue scratch of odd width *w* and height *h*, denoted as $I_l^{w,h}$, $l=1, \ldots, L$; $w=5,7, \ldots, 15$, $h=50, 60, 70$, have been obtained as:

$$\vec{I}_l^{w,h}(i,j)^T = \begin{cases} \vec{I}_l(i,j)^T + [0, s_{w,h}(j)/f, s_{w,h}(j)]^T & if\ (i,j) \in \Omega_l^w \\ \vec{I}_l(i,j)^T & otherwise \end{cases} \qquad (1)$$

where $\vec{I}_l(i,j)^T = [I_l(i,j,1), I_l(i,j,2), I_l(i,j,3)]^T$, $\vec{I}_l^{w,h}(i,j)^T = [I_l^{w,h}(i,j,1), I_l^{w,h}(i,j,2), I_l^{w,h}(i,j,3)]^T$, Ω_l^w denotes the scratch domain, that is the rectangular subset of the image domain of size $N_l \times w$ having as first column the centre column $b=M_l/2$ of the image:

$$\Omega_l^w = \{ (i,j) : i = b, \ldots, b+w-1; j=1, \ldots, N_l \},$$

and $s_{w,h}(j)$ denotes the complete cubic spline interpolating points $(b-1,0)$, $(b+w/2,h)$, $(b+w,0)$.

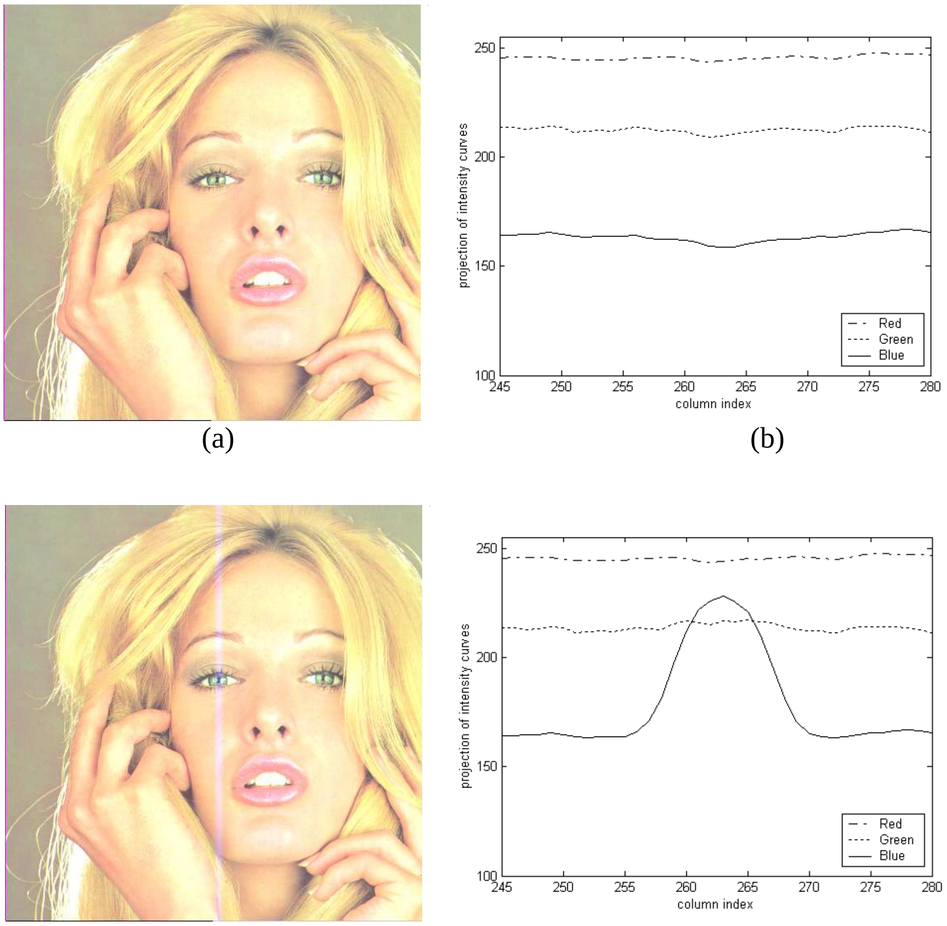
An example of an image $I_l^{w,h}$ artificially corrupted with a blue scratch of width w=15 and height h=70 is given in Fig. 7, together with the horizontal projection of the intensity curves for its three bands; all the other artificially corrupted images $I_l^{w,h}$ are available at web page [10], together with corresponding results obtained with the proposed algorithms.

Knowing *a priori* the scratch mask for such images, we can then apply BSD algorithm to the corrupted images and have an error estimate. For each mask $B_l^{w,h}$ computed with BSD algorithm for the artificially scratched image $I_l^{w,h}$ described in (1), with size $N_l \times M_l$, we count:

- $C_l^{w,h}$= number of *correct detections* (scratch pixels that are included in the computed scratch mask);
- $F_l^{w,h}$= number of *false alarms* (pixels not belonging to the scratch that are included in the computed scratch mask),

and their rates $RC_l^{w,h}$ and $RF_l^{w,h}$ over their respective domains:

- $RC_l^{w,h} = C_l^{w,h} / (N_l \times w)$, $N_l \times w$ being the number of corrupted pixels (i.e. the dimension of the set Ω_l^w);
- $RF_l^{w,h} = F_l^{w,h} / (N_l \times M_l - N_l \times w)$.



(a) (b)

(c) (d)

Fig. 7: Example of artificial blue scratch: (a) original image; (b) horizontal projection of the intensity curves of the three bands of original image; (c) image corrupted with blue scratch of width w=15 and height h=70; (b) horizontal projection of the intensity curves of the three bands of corrupted image.

Given the scratch width *w* and the height *h*, the measures adopted for the objective estimation of BSD algorithm are:

- *mean correct detection rate:* $RC^{w,h} = \frac{1}{L}\sum_{l=1}^{L} RC_l^{w,h}$. Such measure gives values in [0,1]; the higher the value of $RC^{w,h}$, the better the detection result;
- *mean false alarm rate:* $RF^{w,h} = \frac{1}{L}\sum_{l=1}^{L} RF_l^{w,h}$. Such measure gives values in [0,1]; the lower the value of $RF^{w,h}$, the better the detection result.

Values for $RC^{w,h}$ obtained with BSD algorithm applied to images $I_l^{w,h}$ described in (1), varying the scratch width *w* and height *h*, are reported in Fig. 8. Here we can observe that they are generally quite high, even if they tend to decrease increasing the scratch width *w* and decreasing height *h*, in accordance with the increasing difficulty in detecting blue scratches as the width widens and as the height decreases. Corresponding $RF^{w,h}$ values are always close to zero.

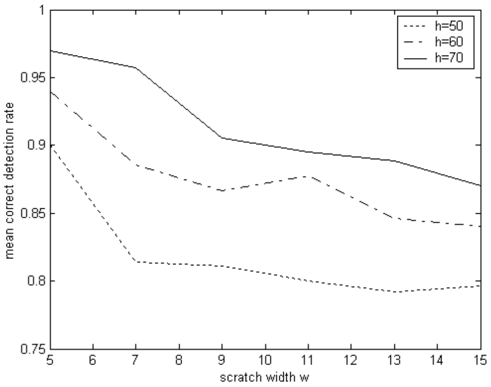
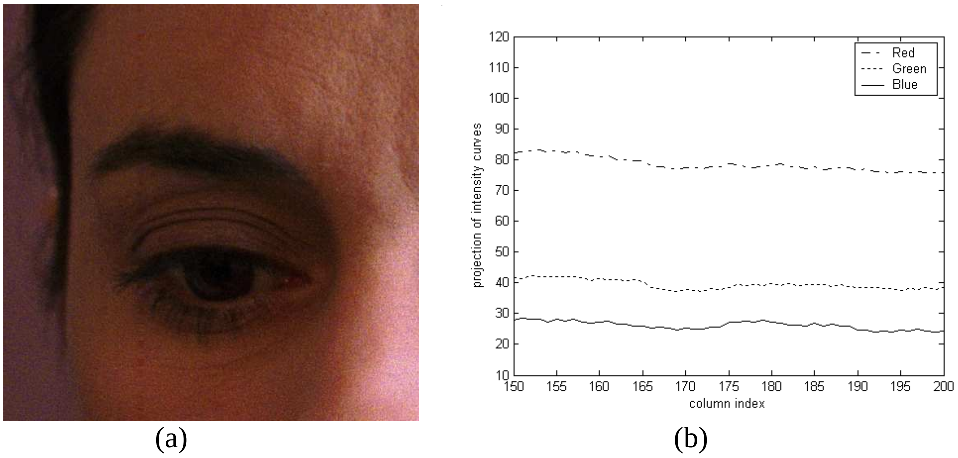


Fig. 8: Error estimates for BSD algorithm applied to images described in (1): mean correct detection rate.

The computational complexity of BSD algorithm, in terms of comparisons and arithmetic operations involved, for an image of size *N×M* affected by a blue scratch of width *w* is O(*N*M*w)*. Just to give an idea, execution times of BSD algorithm, implemented in ANSI C on a Pentium IV, 2GHz, 256Mbytes RAM, for 24 bits RGB colour images of size 256*256, 576*720, and 2048*2880, affected by a blue scratch of width *w* ranging from 5 to 15 pixels are nearly 0.03 s, 0.2 s, and 6.9 s, respectively. We conclude that execution times are quite low for reduced size images; however, they are not sufficiently low for *real time* blue scratch detection in the case of movie resolution images. Parallelisation strategies for BSD algorithm are currently under examination.

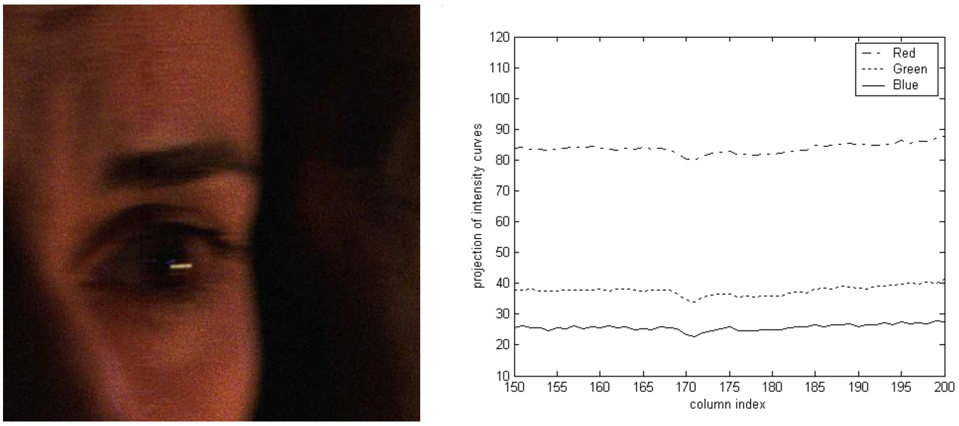
5.2. Evaluation of BSR algorithm

The result of BSR algorithm applied to the naturally corrupted images of Figs. 1, 3, and 4 and to the artificially corrupted image of Fig. 7 is shown in Figs. 9, 10, 11, and 12, respectively, together with the horizontal projection of the intensity curves of their three bands. Here we can observe that BSR algorithm performs in a quite satisfactory way from the subjective visual point of view.



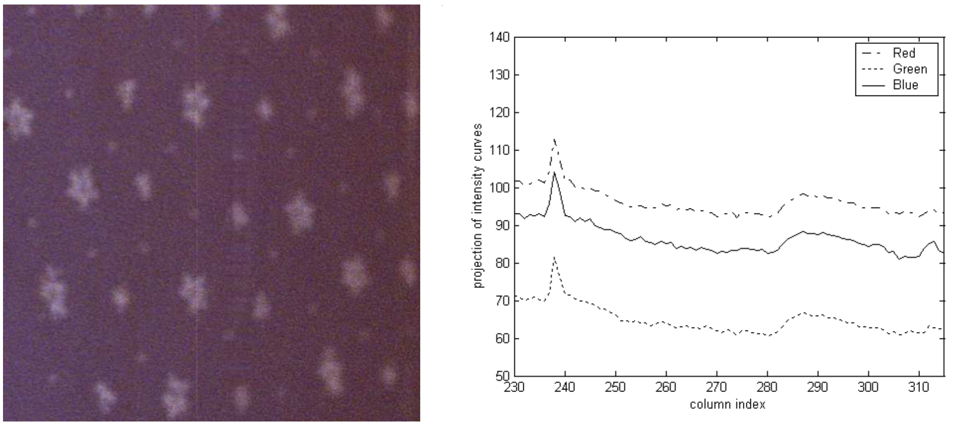
(a) (b)

Fig 9: BSR algorithm for the image of Fig. 1: (a) restored image; (b) horizontal projection of the intensity curves of the three bands of the restored image.



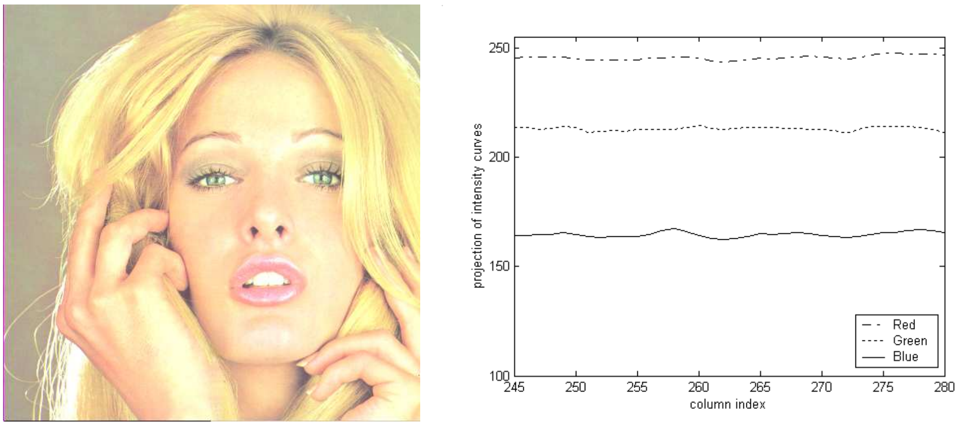
(a) (b)

Fig 10: BSR algorithm for the image of Fig. 3: (a) restored image; (b) horizontal projection of the intensity curves of the three bands of the restored image.



(a) (b)

Fig 11: BSR algorithm for the image of Fig. 4: (a) restored image; (b) horizontal projection of the intensity curves of the three bands of the restored image.



(a) (b)

Fig 12: BSR algorithm for the image of Fig. 7: (a) restored image; (b) horizontal projection of the intensity curves of the three bands of the restored image.

Our aim now is to evaluate the restoration quality attained by BSR algorithm in terms of some objective measure. Therefore, we have again considered the artificially corrupted images $I_l^{w,h}$ of size $N_l \times M_l$ described by (1) used for the evaluation of BSD algorithm. Given the scratch width w and the height h, let be, for $l=1, \ldots, L$:

- o_l the subimage of original image I_l containing only pixels in Ω_l^w,
- r_l the subimage of the restored image $R_l^{w,h}$, obtained with BSR algorithm, containing only pixels in Ω_l^w.

We consider the following objective measures, all computed as the mean over the three bands of each image:

- *MeanMSE*: mean, over the L images, of the Mean Square Error (MSE) between the original and the restored images:

$$MeanMSE = \frac{1}{L}\sum_{l=1}^{L}\frac{1}{N_l \times w}\left\|o_l - r_l\right\|^2,$$

  where ||.|| is intended as vector norm. Such measure gives a nonnegative value; the smaller the value of *MeanMSE*, the better the restoration result;

- *MeanPSNR*: mean, over the L images, of the Peak-to-Noise-Ratio between the original and the restored images obtained considering the MSE:

$$MeanPSNR = \frac{1}{L}\sum_{l=1}^{L}\left(10 * log_{10}\left(\frac{255^2}{\frac{1}{N_l \times w}\left\|o_l - r_l\right\|^2}\right)\right).$$

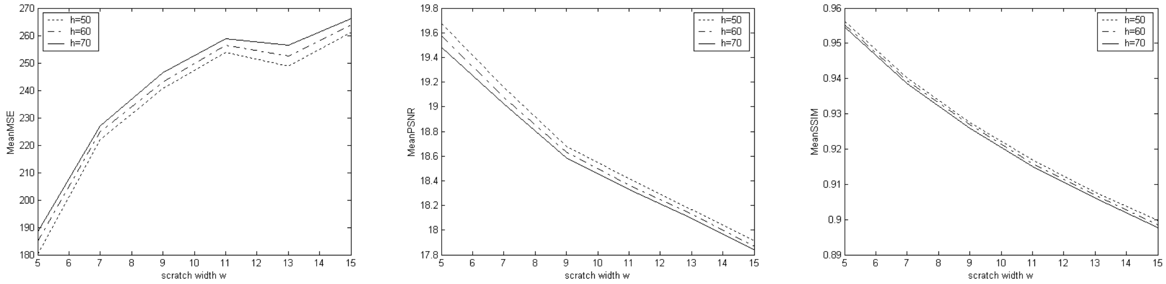
  Such measure gives a nonnegative value; the higher the value of *MeanPSNR*, the better the restoration result;

- *MeanSSIM*: mean, over the L images, of the Structural Similarity Index [33] applied to the original and the restored images:

$$MeanSSIM = \frac{1}{L}\sum_{l=1}^{L}\frac{\left(2 * \mu_{o_l} * \mu_{r_l} + C_1\right)\left(2 * \sigma_{o_l r_l} + C_2\right)}{\left(\mu_{o_l}^2 + \mu_{r_l}^2 + C_1\right)\left(\sigma_{o_l}^2 + \sigma_{r_l}^2 + C_2\right)},$$

  where $C_1=(K_1*A)^2$, $C_2=(K_2*A)^2$, $K_1=0.01$, $K_2=0.03$, and $A=255$. Such measure gives values in [0,1]; the higher the value of *MeanSSIM*, the better the restoration result.

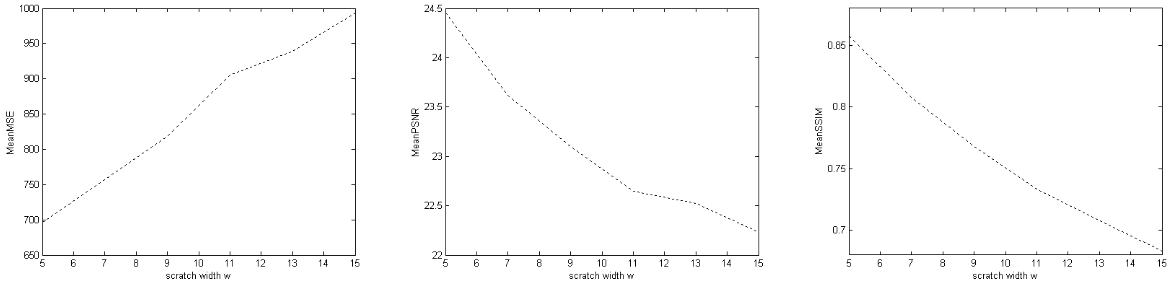
Results in terms of the described measures obtained by BSR algorithm varying the scratch width w and height h are reported in Fig. 13, and show that statistical properties of the original images are quite well restored. Moreover, it can be observed that results obtained with all the considered measures show lower accuracy increasing the scratch width *w* and height *h*, in accordance with the increasing reconstruction difficulty as the reconstruction area widens and as the scratch contrast grows.



(a) (b) (c)

Fig 13: Error measures for BSR algorithm applied to images described in (1): (a) MeanMSE; (b) MeanPSNR; (c) MeanSSIM.

Such results have also been compared with analogous results obtained with an implementation of the inpainting algorithm (*missing data* approach) presented in [4], shown in Fig. 14. Here we can observe that all the considered error measures attain values worse than those obtained by BSR algorithm.

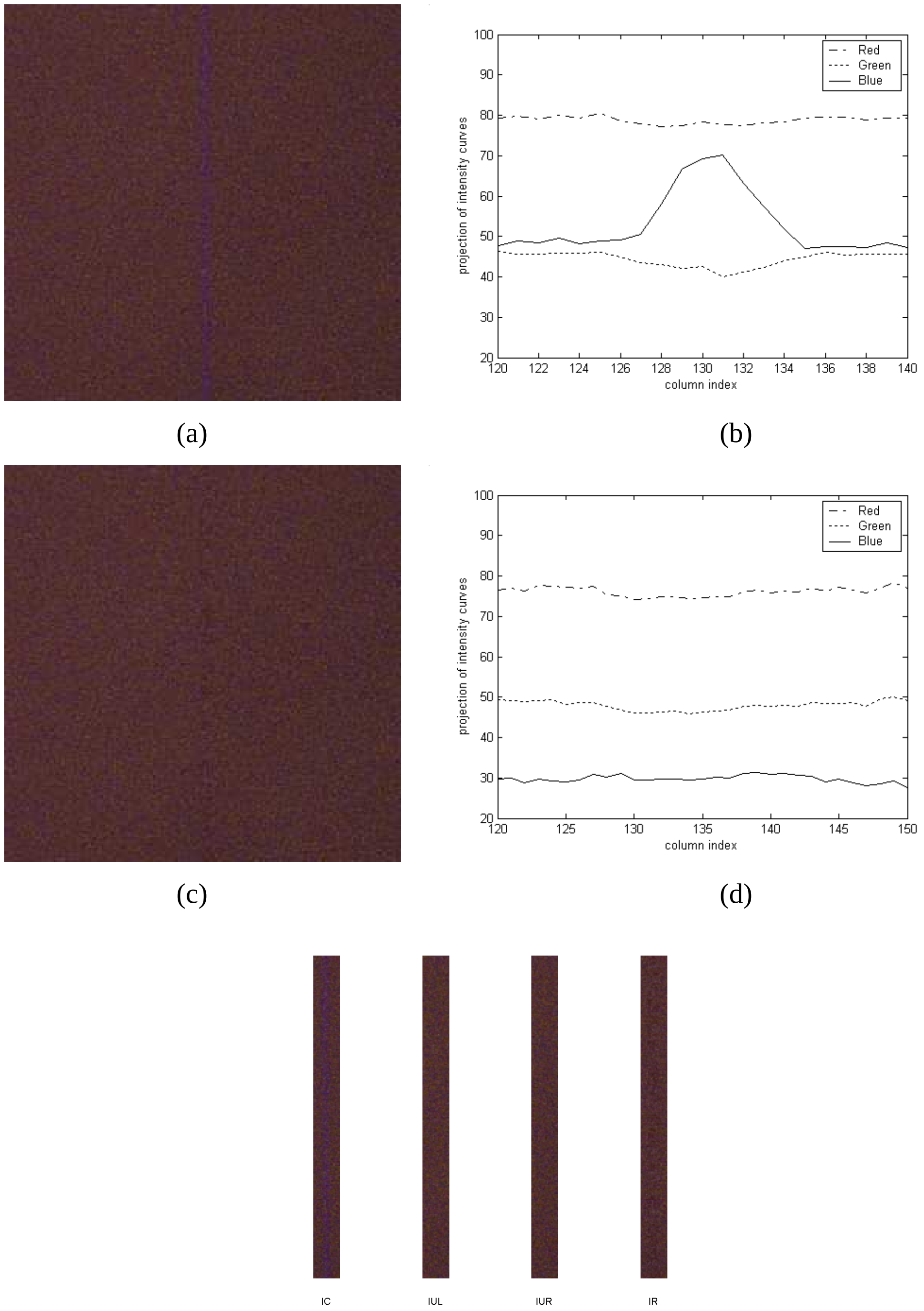


(a) (b) (c)

Fig 14: Error measures for the inpainting algorithm presented in [4] applied to images described in (1): (a) MeanMSE; (b) MeanPSNR; (c) MeanSSIM.

Conscious that, due to the specific features of blue scratches, the defect cannot be perfectly simulated on an uncorrupted image, we performed also different accuracy measurements. Having at our disposal almost uniform real images affected by blue scratches (reported in Figs. 15 and 16), we have taken the above measures on subblocks of such images. Specifically, for the image of Fig. 15-(a) showing a blue scratch of average width 9 (from column 127 to column 135), we have considered as corrupted image, I_C, the subimage of the original image containing a block of columns that include the blue scratch (from column 121 to column 141), and we have considered two uncorrupted images, I_{UL} and I_{UR} , the first containing a block of uncorrupted columns on the left of I_C (from column 100 to 120) and the second containing a block of uncorrupted columns on the right of I_C (from column 142 to 162). Applying BSR algorithm to the corrupted image I_C, we have obtained the restored image I_R. Sub-images I_C, I_{UL},I_{UR}, and I_R, of the image of Fig. 15-(a) are reported in Fig.

15-(e). The mean, the standard deviation, and the L_2 norm for the corrupted image I_C, for the uncorrupted images I_{UL} and I_{UR} and for the restored image I_R are compared and reported in Table 1. The results confirm that BSR algorithm performs quite well for blue scratches of standard width.



(a)

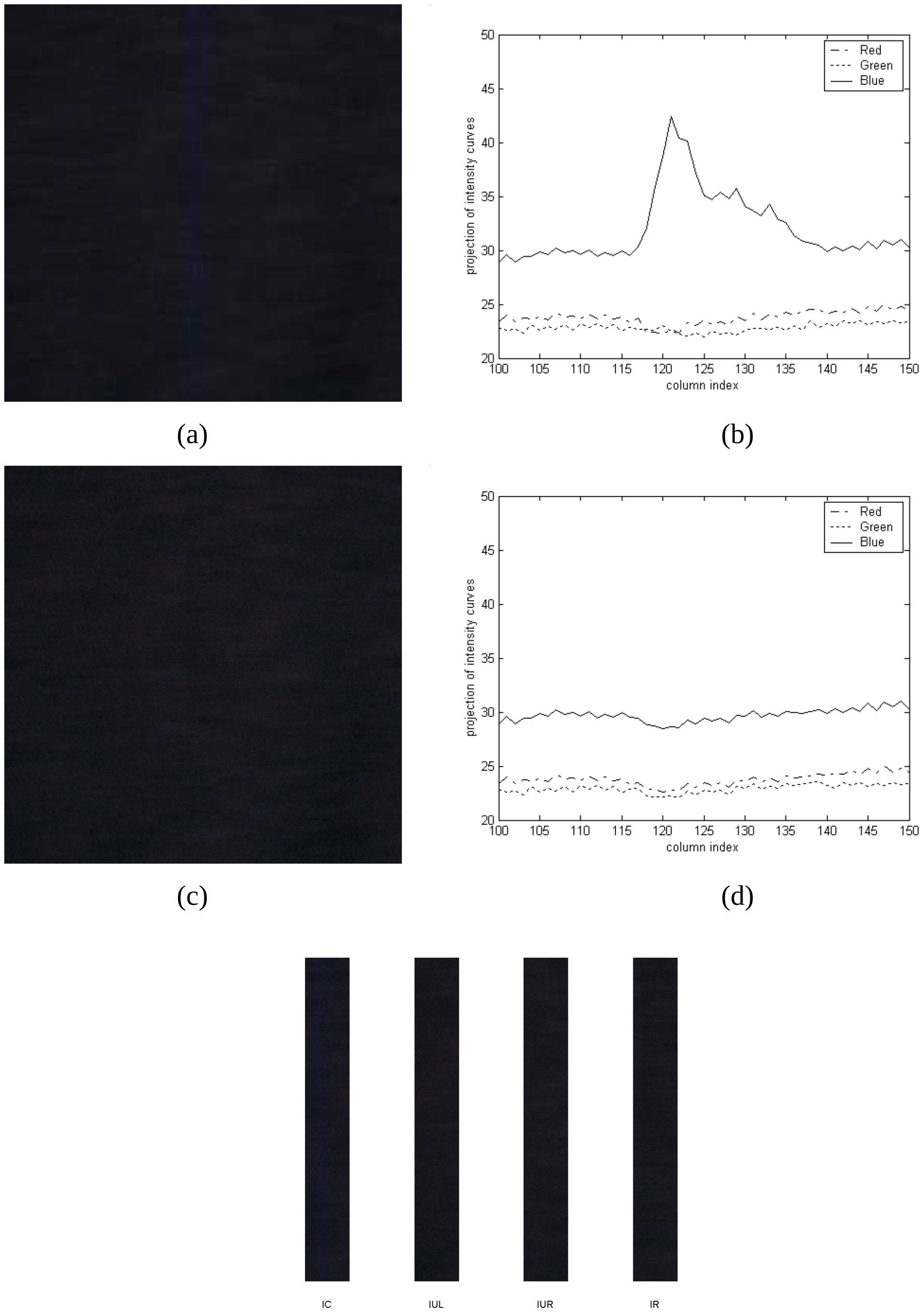
(b)

(c)

(d)

(e)

Fig 15: Example of a blue scratch on a uniform background: (a) original image; (b) horizontal projection of the intensity curves of the three bands of the original image; (c) restored image; (d) horizontal projection of the intensity curves of the three bands of the restored image; (e) subimages considered for the measures reported in Table 1.



(a)

(b)

(c)

(d)

(e)

Fig 16: Example of a wide blue scratch on a uniform background: (a) original image; (b) horizontal projection of the intensity curves of the three bands of the original image; (c) restored image; (d) horizontal projection of the intensity curves of the three bands of the restored image; (e) subimages considered for the measures reported in Table 2.

Sub-image	Mean	Std. Dev.	L_2 norm
I_C	58.72	8.04	4352
I_{UL}	58.22	6.96	4302
I_{UR}	57.17	6.92	4225
I_R	57.25	7.03	4232

Table 1: Mean, standard deviation and L_2 norm for the corrupted image I_C, for the uncorrupted images I_{UL} and I_{UR}, and for the restored image I_R reported in Fig. 15.

Sub-image	Mean	Std. Dev.	L_2 norm
I_C	26.48	3.17	2525
I_{UL}	24.64	2.87	2349
I_{UR}	26.07	2.63	2481
I_R	25.38	2.57	2415

Table 2: Mean, standard deviation and L_2 norm for the corrupted image I_C, for the uncorrupted images I_{UL} and I_{UR}, and for the restored image I_R reported in Fig. 16.

Analogous measures for the almost uniform image of Fig. 16-(a) are reported in Table 2. In this case the average scratch width is 23 pixels; the corrupted image I_C , containing a block of columns of the image including the blue scratch (from column 111 to column 145), and the two uncorrupted images I_{UL} and I_{UR}, containing the block from column 76 to 110 and from column 146 to 180, respectively, are shown in Fig. 16-(e), together with the restored image I_R obtained applying BSR algorithm to I_C. The results confirm that BSR algorithm performs quite well also for very large blue scratches.

The computational complexity of BSR algorithm is quite low, including a number of comparisons linearly proportional to the size of the image and a number of arithmetic operations linearly proportional to the number of rows of the image and the scratch width. Execution times of BSR algorithm, in ANSI C on a Pentium IV, 2GHz, 256Mbytes RAM, for 24 bits RGB colour images of size 256*256, 576*720, and 2048*2880, affected by a blue scratch of width *w* ranging from 5 to 15 pixels are nearly 0.002 s, 0.01 s, and 0.55 s, respectively. Therefore, we can conclude that execution time is generally sufficiently low for *real time* blue scratch removal, even for movie resolution images.

6. Conclusions

We considered the problem of detecting and removing blue scratches from digital image sequences. In particular, we analysed in detail the specific features of such kind of scratches and proposed a detection method and a removal method that strongly rely on these features. A thorough analysis of the algorithms accuracy, accompanied by several numerical experiments carried out on both naturally and artificially corrupted images, show that the proposed detection and removal algorithms produce satisfying results. The performance of the algorithms, in terms of execution times, is quite good for TV resolution images; however, for the case of movie resolution images the detection algorithm does not allow real time computation, requiring execution times in the order of tens of seconds. Parallelisation strategies for the detection algorithm are currently under examination.

7. Acknowledgements

The authors would like to express their gratitude to the anonymous referees for their useful suggestions. This work has been partially supported by the *Regional Competence Centre for the Development and Transfer of Innovation Applied to Cultural and Environmental Heritage* (*INNOVA*) funded by Regione Campania, Italy.

References

[1] A. Anzalone, A. Machì, A Method for Accurate Detection of Linearly Scratched Areas in Motion Pictures, Proceeding of IASTED-VIIP01, Spain (2001) 565-570.

[2] C. Ballester, M. Bertalmio, V. Caselles, G. Sapiro, J. Verdera, Filling-in by joint interpolation of vector fields and grey levels, IEEE Transactions on Image Processing 10 (2001) 1200-1211.

[3] C. Ballester, V. Caselles, J. Verdera, Disocclusion by joint interpolation of vector fields and grey levels, SIAM Journal Multiscale Modelling and Simulation 2 (2003) 80-123.

[4] R. Bornard, E. Lecan, L. Laborelli, J.-H. Chenot, Missing data correction in still images and image sequences, Proceedings of ACM Multimedia (2002) 355-361.

[5] T. Bretscheneider, O. Kao, P.J. Bones, Removal of Vertical Scratches in Digitised Historical Film Sequences Using Wavelet Decomposition, Proceedings of Image and Vision Computing Conference, Hamilton, New Zealand (2000) 38-43.

[6] V. Bruni, D. Vitulano, A Generalized Model for Scratch Detection, IEEE Transactions on Image Processing 13 (2004) 44-50.

[7] Color Texture Analysis, Institut fur Computervisualistik, Universitat Koblenz-Landau, http://www.uni-koblenz.de/FB4/Institutes/ICV/AGPriese/Research/Color%20Texture%20Analysis.

[8] Computer Vision Laboratory, Computer Science Department, University of Massachussets, http://vis-www.cs.umass.edu/vislib/Houses/IM4/images.html.

[9] E. Decenciere Ferrandiere, Restauration automatique de films anciens, PhD Thesis, Ecole Nationale Superieure des Mines de Paris (1997).

[10] DFR Laboratory, ICAR-CNR, Naples, www.na.icar.cnr.it/~maddalena.l/DFRLab/BlueScratches.html.

[11] A.A. Efros, W.T. Freeman, Image Quilting for texture synthesis and transfer, Proceedings of 28th Annual Conference on Computer Graphics and Interactive Techniques (2001) 341-346.

[12] L. Joyeux, O. Buisson, B. Besserer, S. Boukir, Detection and Removal of Line Scratches in Motion Picture Films, Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, Fort Collins, Colorado (1999) 548-553.

[13] L. Joyeux, Reconstruction de sequences d'images haute resolution. Application a la restauration de films cinematographiques, PhD Thesis, Universitè de La Rochelle (2000).

[14] 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.

[15] L. Joyeux, S. Boukir, B. Besserer, Tracking and MAP reconstruction of line scratches in degraded motion pictures, Machine Vision and Applications 13 (2002) 119-128.

[16] O. Kao, J. Engehausen, Scratch Removal in Digitised Film Sequences, Proceedings of International Conference on Imaging Science, Systems, and Technology, CSREA Press (2000) 171-179.

[17] A.C. Kokaram, R. Morris, W. Fitzgerald, P. Rayner, Detection/Interpolation of Missing Data in Image Sequences, IEEE Transactions on Image Processing 4 (1995) 1496-1519.

[18] A.C. Kokaram, Motion Picture Restoration: Digital Algorithms for Artefacts Suppression in Archived Film and Video, Springer-Verlag (1998).

[19] A.C. Kokaram, A statistical framework for picture reconstruction using AR models, Proceedings of European Conference of Computer Vision, Workshop on Statistical Methods for Time Varying Image Sequences (2002) 73-78.

[20] A.C. Kokaram, Parametric texture synthesis for filling holes in pictures, Proceedings of IEEE International Conference in Image Processing (2002) 325-328.

[21] A. Machì, F. Collura, F. Nicotra, Detection of irregular linear scratches in aged motion picture frames and restoration using adaptive masks, Proceedings of IASTED International Conference on Image Processing (2002) 254-259.

[22] L. Maddalena, Efficient Methods for Scratch Removal in Image Sequences, Proceedings of 11th International Conference on Image Analysis and Processing, IEEE Computer Society (2001) 547-552.

[23] S. Masnou, J.-M. Morel, Level-lines based disocclusion, Proceedings of 5th IEEE International Conference on Image Processing 3 (1998) 259-263.

[24] R.D. Morris, Image Sequence Restoration Using Gibbs Distributions, PhD Thesis, University of Cambridge (1995).

[25] R.D. Morris, W.J. Fitzgerald, A.C. Kokaram, A sampling based approach to line scratch removal from motion picture frames, Proceedings of IEEE International Conference on Image Processing 1 (1996) 801-804.

[26] L. Rosenthaler, A. Wittmann, A. Gunzl, R. Gschwind, Restoration of Old Movie Films by Digital Image Processing, Proceedings of IMAGÈCOM 96, Bordeaux, France (1996) 20-22.

[27] T. Saito, T. Komatsu, T. Ohuchi, T. Seto, Image Processing for Restoration of Heavily-Corrupted Old Film Sequences, Proceedings of International Conference on Pattern Recognition, Barcellona, IEEE Computer Society (2000) 17-20.

[28] G. Sapiro, Image inpainting, SIAM News 35 (2002).

[29] J. Shen, Inpainting and the fundamental problem of image processing, SIAM News 36 (2003).

[30] D. Tegolo, F. Isgrò, A Genetic Algorithm for Scratch Removal in Static Images, Proceedings of International Conference on Image Analysis and Processing, IEEE Computer Society (2001) 507-511.

[31] USC-SIPI Image Database, Electrical Engineering Department, Signal & Image Processing Institute, University of Southern California, http://sipi.usc.edu/database.

[32] J. Verdera, V. Caselles, M. Bertalmio, G. Sapiro, Inpainting surface holes, Proceedings of IEEE International Conference on Image Processing (2003), 14-17.

[33] Z. Wang, L. Lu, A.C. Bovik, Video quality assessment based on structural distortion measurement, Signal Processing: Image Communication 19 (2004) 121-132.