Hindawi Publishing Corporation
EURASIP Journal on Image and Video Processing
Volume 2010, Article ID 680429, 11 pages
doi:10.1155/2010/680429

Research Article

Multidirectional Scratch Detection and Restoration in Digitized Old Images

E. Ardizzone, H. Dindo, and G. Mazzola (EURASIP Member)

Dipartimento di Ingegneria Informatica (DINFO), Università degli Studi di Palermo, Viale delle Scienze, Building 6, 90128 Palermo, Italy

Correspondence should be addressed to E. Ardizzone, ardizzon@unipa.it

Received 26 April 2010; Revised 19 September 2010; Accepted 7 October 2010

Academic Editor: Sebastiano Battiato

Line scratches are common defects in old archived videos, but similar imperfections may occur in printed images, in most cases by reason of improper handling or inaccurate preservation of the support. Once an image is digitized, its defects become part of that image. Many state-of-the-art papers deal with long, thin, vertical lines in old movie frames, by exploiting both spatial and temporal information. In this paper we aim to face with a more challenging and general problem: the analysis of line scratches in still images, regardless of their orientation, color, and shape. We present a detection/restoration method to process this defect.

1. Introduction

Scratches are typical damages of old movie films and occur as dark or bright vertical lines which run all over the frames of a video. They are typically caused by the lost of the emulsion of the film surface, due to contact with mechanical parts of film projector or other devices in the film development process. In our work we focused on defects of old damaged photos. Old photographic prints may present several types of defects, caused by inaccurate handling and/or store of the original image, or by chemical factors, or by decomposition of the support. If the knowledge of a degradation origin is essential for defect analysis on the physical support, different defects may look similar once the document is digitized and could be described and removed by similar underlying processes. Several works rely on the damage analysis and restoration of digitized pictures: cracks and craquelures [1, 2], water blotches and foxing [3], and fading [4]. For a complete taxonomy of the defects in old photos see [5].

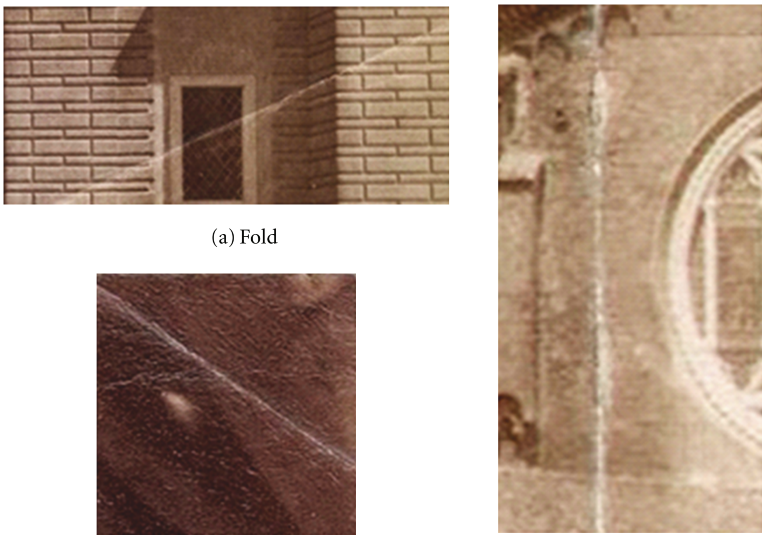
In this paper we deal with scratches, defects which may be originated by several physical causes, but that share some common features (Figure 1). In the next two sections we will discuss the differences between scratches in videos and in photos and the different approaches to analyze them.

2. Scratches in Videos

The restoration of old damaged videos had been widely studied in the last two decades. When digitized, movie films can be processed by digital postprocessing techniques in order to reduce or remove unwanted artifacts (scratches, spots, etc.). A typical restoration process is made of two steps: detection and restoration.

Most of the approaches use both spatial (from the current frame) and temporal (from adjacent frames) information to detect the presence of line scratches into the video.

All the state-of-the-art methods for videos dealt with a very simple kind of defect: long vertical dark (or bright) scratches, with constant color, width and orientation. Although some authors used a spatiotemporal point of view to process scratches [6–8], some papers proposed static approaches, using information from a single frame. Therefore these methods can be used to process also vertical scratches in still images. Kokaram [9] proposed a 2-dimensional autoregressive model for scratch detection and removal, without using information from adjacent frames. Bretschneider et al. [10] proposed a technique based on wavelet decomposition. Bruni and Vitulano [11] generalized Kokaram's model for scratch detection on the hypothesis that a scratch is not purely additive on a given image. Tegolo and


(a) Fold

(b) Abrasion

(c) Tear

Figure 1: A comparison between three scratches, which have similar digital aspects but originated by different causes.

Isgrò [12] approach is based on the analysis of the statistics of the image grey levels.

3. Scratches in Still Images

The problem of processing scratches is much harder to deal in still images than in old movies, at least for two reasons: first, scratches in still images may have any possible orientation, color, and width while scratches in video have very specific features (see Figure 2); second, no temporal information can be used to process them.

The goal of our work is to find a general approach to the problem, to detect and restore scratches regardless of their specific features. Our general "model" of a scratch is a long thin line which runs along all the image with any orientation and color, but which may also have small variations in width and direction, some little interruptions, but no branches. In most cases, scratches in damaged photos are due to the deterioration of the emulsion or the support of the film. Thus, scratches cannot be considered as "digital" defects, as they come from the digitization of printed documents. Nevertheless, once a photo is digitized, scratches become part of the digital photo and can be digitally processed. The first step is to detect the orientation and the position of the scratch into the image. Then our method labels pixels which are part of the detected line scratch, tracing its contour. To our knowledge, no works in literature deal with the problem of "tracking" scratches, as they typically stop detection when they find the position of the (vertical) line. Last, scratches are restored and lost information recovered. Present work expands over our previous works in quasihorizontal scratch restoration [13] and multidirectional scratch detection [14].

4. Multidirectional Detection

The approach presented in [13] is based on a bandpass filtering, to enhance the horizontal components of the image, and a Hough transformation, to detect candidate line scratches. Therefore, scratches which have slopes with more than 30° cannot be detected. This is not enough to achieve our goals. Our extended solution can be divided into three steps: preprocessing, line detection, and contour drawing.

4.1. Preprocessing

4.1. Preprocessing. The preprocessing step aims to enhance image features along a set of chosen directions. First, image is grey-scaled and filtered with a sharpening filter (we subtract from the image its local-mean filtered version), thus eliminating the DC component. Tests showed that this solution makes our approach independent of the color of the scratch, and it helps the next steps in detecting its direction.

Next step employs a bank of oriented bandpass filters. Kass and Witkin [15] affirmed that a line is 90° shifted in the frequency domain and suggest the use of a bandpass filter:

$$H(u,v) = \frac{1}{1 + 0.414 \cdot \left(\sqrt{(u^*/D_h) + (v^*/D_v)}\right)^{2n}}, \quad (1)$$

where D_h and D_v are the two cutoff frequencies while u^* and v^* are the translated and rotated frequency coordinates:

$$\begin{aligned} tx &= \text{center} \cdot \cos(\theta), \\ ty &= \text{center} \cdot \sin(\theta), \\ u^* &= \cos(\theta) \cdot (u + tx) + \sin(\theta)(u + ty), \\ v^* &= -\sin(\theta) \cdot (u + tx) + \cos(\theta)(u + ty), \end{aligned} \quad (2)$$

so that the center of the subband has t_x, t_y coordinates and it is rotated with the same angle. The subbands must be symmetric with respect to the origin; so the angle θ must be shifted of 90°. The filter order n can control the slope of the subband, so that we chose $n = 4$ to concentrate the filtering in a precise zone of the spectrum. This explains the use of this bandpass filter instead of a Gabor one.

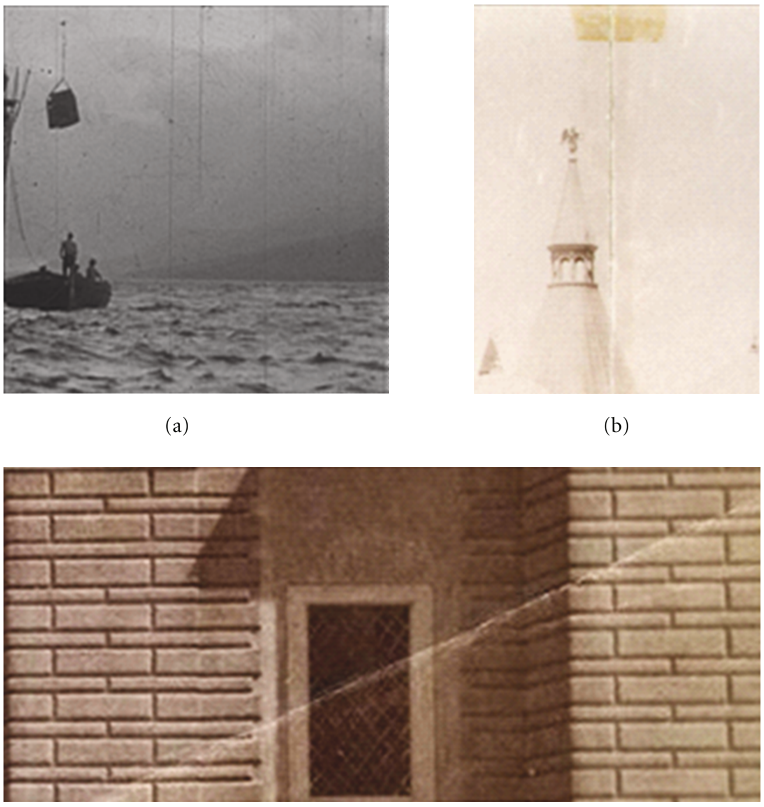


FIGURE 2: Three examples of images with scratches: a frame of an old movie (a) and two crops from old photos ((b) and (c)).

We selected 12 not overlapping filters, to analyze 12 different directions, rotated with respect to 15° each other. A homomorphic filter is then applied to enhance lines and to produce a dark uniform background. Finally a threshold is applied to obtain a binary mask. The threshold value is computed as mean plus standard deviation of the image intensity (see Figure 3(b)).

4.2. Line Direction Detection. After pre-processing, we apply the Hough transform to the 12 output binary images, in order to extract all the relevant lines. Hough transform is used to represent an image in a parameter space, in this case slope and intercept of a generic line. Usually, the parameter space is subdivided into a number of accumulator cells in order to reduce the computational complexity. Each pixel is processed and a counter in the accumulator cell is incremented. We observed that since each binary image is the output of a directional filtering process, the Hough transform will give relevant information only in a neighborhood of that direction (shifted by 90° with respect of the input image). Therefore, for each direction, only this information is useful, and we saved it as columns in a data matrix. We then search for the maximum value which represents our first line scratch, and considered as a reference to search for other potential scratches. We suppose that other scratches, probably originated by the same physical causes, have in the data matrix lower but similar values than those of the first one. Furthermore we observed that, since the Hough transform response depends on the number of points which are accumulated along a specific direction, some directions

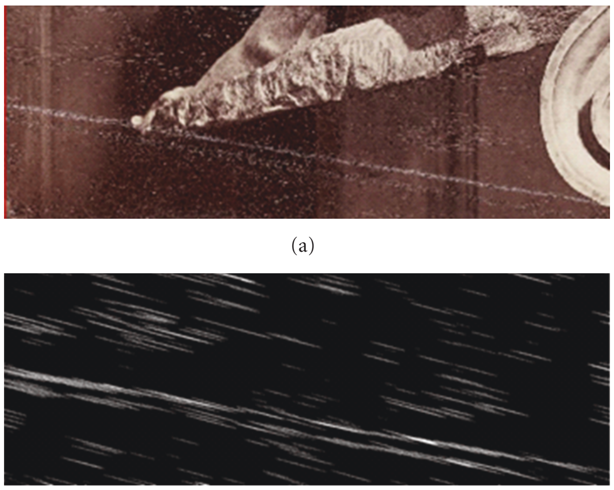


FIGURE 3: Input image (a) and one of the output of the prefiltering step (b) (the direction of the diagonal scratch shifted by 90°).

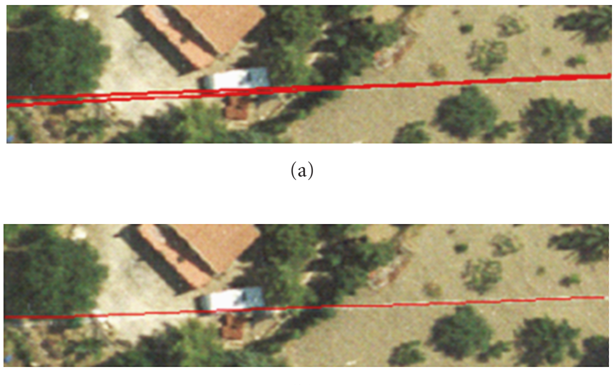


Figure 4: The effect of resetting, in the data matrix, the values in a neighborhood of the scratch candidate value.

(typically the diagonal ones) are advantaged with respect to the others, because a longer line can include a greater number of points. Therefore we consider the next maximum value in the data matrix and we compare it with a threshold, which depends both on the reference value, and on the length of the longer possible line with the same direction. Whenever a new scratch is found, its value, all the values in a neighborhood are set to zero. This solution is applied to avoid the detection of multiple overlapping scratches (Figure 4), which can be revealed since the same set of points can be included in similar but distinct lines. If the candidate value is lower than a fixed threshold, independent of the image, the detection step ends (see Figure 5).

4.3. Contour Drawing. Contour-drawing step requires some user intervention. We propose to the user a set of possible scratches, and he has to select one of the candidates for the next steps in the scratch analysis process. Note that scratches in old photos may be originated by heterogeneous causes, so that they can present interruptions, irregularities in width, slope, and color. Digitization adds a further problem: when digitized, defects of the paper become part of the image. During the acquisition of a printed image, typically an interpolation technique is applied; so the intensity value of the pixels varies according to the surrounding background. Therefore, pixels in the same scratch can have very different intensity values, if the scratch passes across darker and brighter areas.

The contour drawing step can be divided into two substeps: scratch "core" detection and region growing. The first step aims to find the skeleton of the scratch, the second one to find its local thickness.

4.3.1. Core Detection. The aim of the core detection is to find the backbone of the scratch in the most accurate way. For this step we work with lines with slope in $[-45^\circ, 45^\circ]$. If the line slope is outside this interval, the image is 90° shifted in order to bring the scratch line into this case.

This step needs for a starting reference point to proceed in the analysis, since the Hough transform returns only the direction and the position of the scratch line. The reference point is manually chosen by the user, who decides which is the most significant one for the selected scratch. Starting from this initial point, our method searches for the 5 nearest pixels (see Figure 6) to include the "most similar" ones into the scratch core (note that the image could have been rotated). The similarity function depends on two factors: a "global" correlation factor, with respect to reference point,

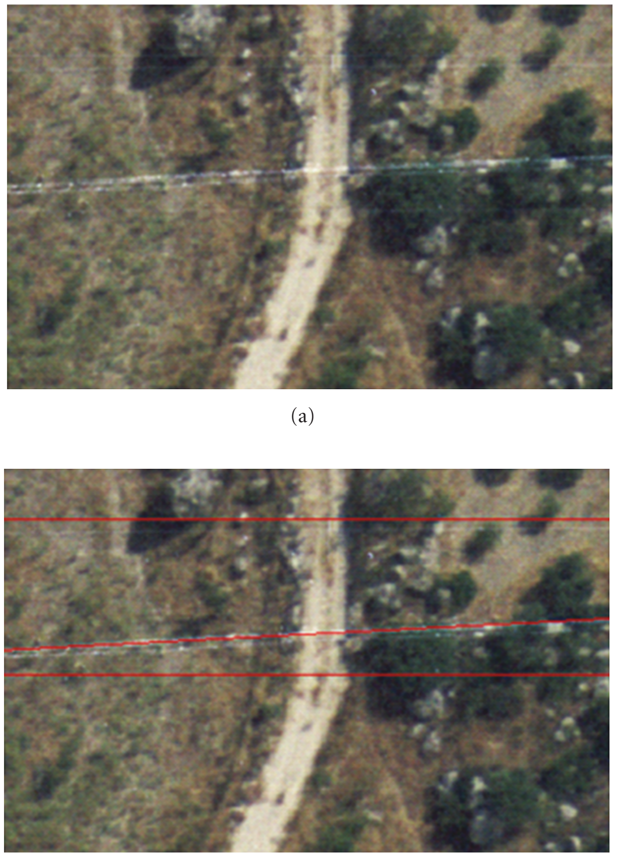


Figure 5: Multiple scratch detection. Input image (a), and three correctly detected scratches (b). Some scratches are not detected since they are "covered" by close ones.

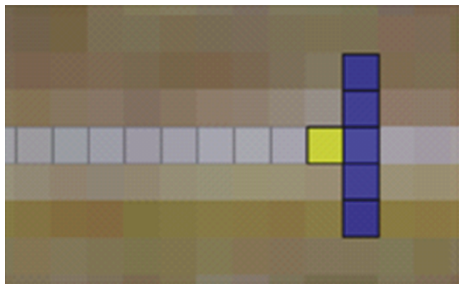


Figure 6: The five nearest pixels (in blue) to the reference point (in yellow), rightward. The same procedure is repeated leftward.

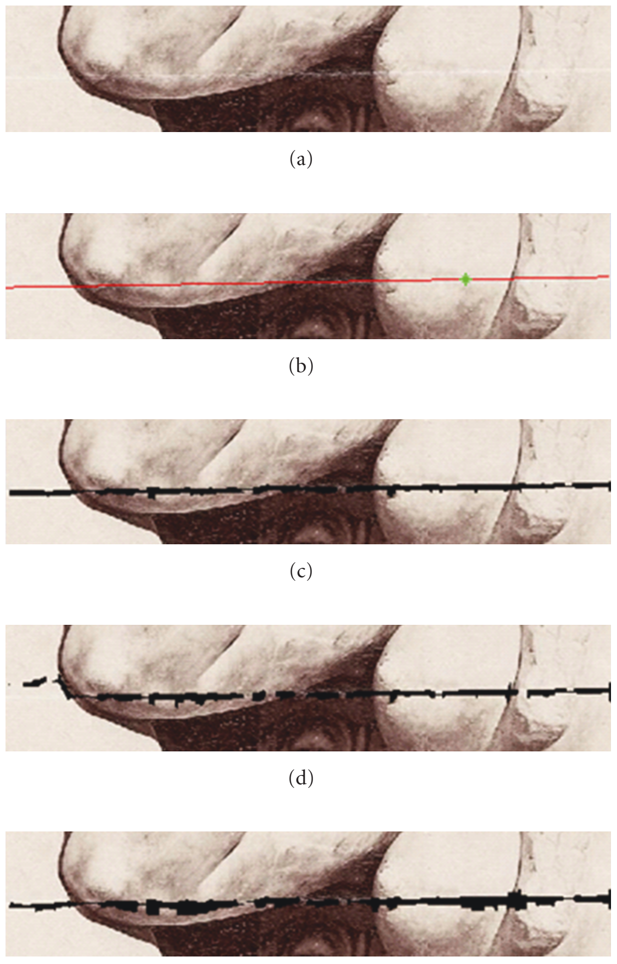


Figure 7: Original image, with a scratch (a); detected scratch and starting point (b); scratch contour with default parameters: $w_1 = 0.75$, $w_2 = 0.25$, $w_1' = 0.85$, and $w_2' = 0.15$ (c); scratch contour with $w1 = 0.5$ and $w2 = 0.5$ (d); scratch contour with $w_1' = 0.5$ and $w_2' = 0.5$ (e).

and a “local” correlation factor between the candidate and the last included pixel:

$$s_c(p_i, k) = w_1 \cdot c_g\left(p_r, p_i^k\right) + w_2 \cdot g(p_i, d_s) \cdot c_l\left(p_s^{k-1}, p_i^k\right). \tag{3}$$

(i) $s_c(p_i, k)$ is the similarity function for the core detection, evaluated in the candidate point p_i of the column k.

(ii) c_g is the “global” correlation value, computed between two windows, centered, respectively, in the starting point p_r and in the candidate point p_i of the column k;

(iii) c_l is the “local” correlation value, computed between two windows, centered, respectively, in the last included point p_s for the previous column, and in the candidate point p_i of the column k.

(iv) $g(p_i, d_s)$ is a Gaussian function, evaluated in p_i, with a peak in correspondence with the detected scratch direction d_s and a constant standard deviation;

(v) w_1 and w_2 are fixed weights, set, respectively, to 0.75 e 0.25. The influence of these values on the results is shown in Figure 7. If we increase the “local” weight, we increase the probability to deviate from the principal direction, due to possible local maxima (note the left part of Figure 7(d)). The chosen values give best results, within our dataset.

For each column we select the point, between the 5 candidates, with the maximum similarity value. If this value is lower than a fixed threshold (set by experiments to 0.2), no pixels are included into the core, but our method continues searching for other points along the direction d_s, until the end of the image.

The global factor is used to find all the pixels, which are similar to the reference pixel, along the scratch direction. On the other hand, the local factor makes this method flexible to follow little changes in slope. Moreover, points can be included into the core in spite of possible interruptions along the scratch main direction. It is clear that the choice of a “good” reference point is critical: starting from a point that is along the scratch direction, but is not part of the scratch, makes the core detection process work incorrectly.

4.3.2. Region Growing. In this phase we search for all the pixels which belong to the scratch, orthogonally with respect to the core direction. The approach is similar to that used in the core detection: we include pixels upward and downward instead of rightward and leftward.

The similarity measure is like that in (3). The main difference, in addition to the search direction, is the Gaussian function, which has its peak on the pixels of the scratch core, and a variance which depends on the last computed scratch thickness:

$$s_t\left(p_j, k\right) = w_1 \cdot c_g\left(p_r, p_j^k\right) + w_2 \cdot g\left(p_j, \mathrm{var}^k\right) \cdot c_l\left(p_{j\pm1}^{k}, p_j^k\right), \tag{4}$$

$$\mathrm{var}^k = \max\left(w_1' \cdot \mathrm{var}^{k\pm1} + w_2' \left(t_s^{k\pm1}\right)^2, k_{\min}\right). \tag{5}$$

(i) $s_t(p_j, k)$ is the similarity function for the thickness detection, evaluated in p_j in the column k.

(ii) w_1' and w_2' are two fixed weights, set by experiments to 0.85 and 0.15. The influence of these values on the results is shown in Figure 7. If we increase the “local” weight, we increase the probability to include “good” pixels into the scratch to be restored (Figure 7(e))

(iii) c_g, c_l, w_1, and w_2 are the same of (3).

(iv) p_{j+1} (p_{j-1}) is the next upper (lower) pixel to be examined in the same column of p_j.

(v) $g(p_j, \mathrm{var}^k)$ is a Gaussian function, evaluated in p_j, with a peak in correspondence with the pixel in column k of the scratch core, and a variance which is shown in (5).

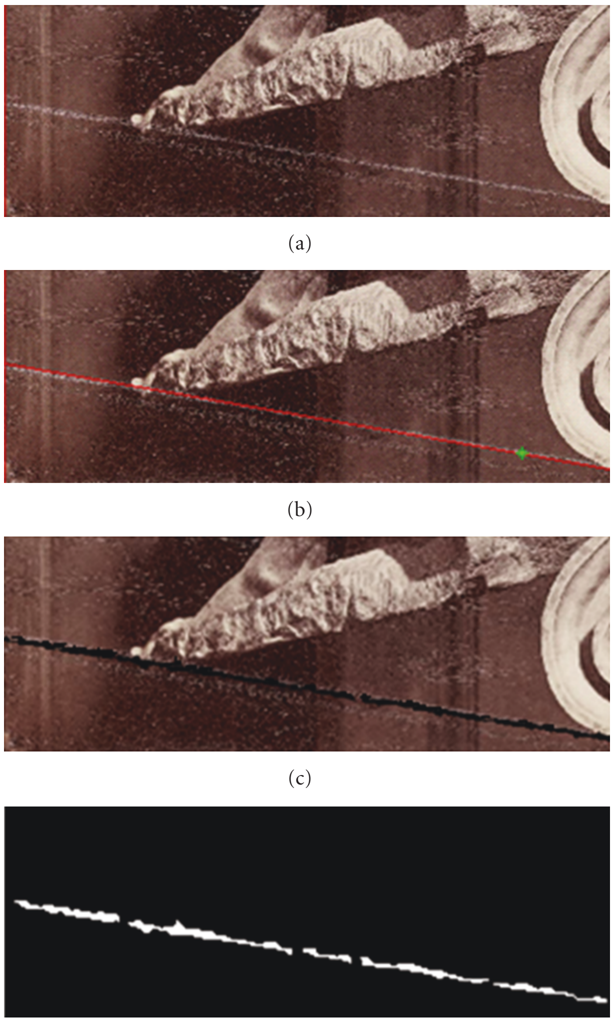

(a)
(b)
(c)
(d)

Figure 8: A complete detection process: the damaged image (a), the detected line (b) and the starting point (the green), the detected scratch with its contour (c) and the binary mask. Note that the detected mask is interrupted in correspondence to the scratch gaps of the input scratch.

(vi) var^k and var^{k+1} (or var^{k-1}) are the variances of the Gaussian function in the column k and $k+1$ (or $k-1$), respectively; $t_s^{k+1}(t_s^{k-1})$ is the thickness for the column $k+1(k-1)$.

(vii) $k_{\min}$ is the minimum admitted value for the variance.

Our method, for each column, stops when the similarity value of the next candidate pixel (upward or downward) is lower than a fixed threshold (set to 0.1). The global factor makes the proposed method able to find all the pixels which are similar to the reference pixel, along the direction that is orthogonal to the core. The local factor and the adaptable variance solution are useful to detect changes in the scratch thickness. The final step of the detection phase is to create a binary image showing the result of the labeling process. Figure 8 shows an example of a complete scratch detection process. Note that (Figure 8(d)) the detected mask breaks in correspondence with the scratch interruptions in the input image.

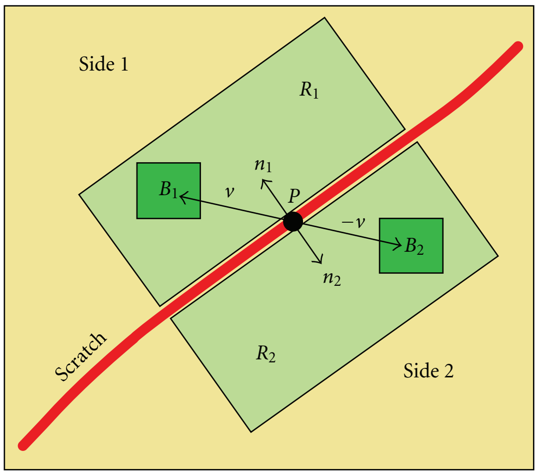


Figure 9: The estimation of the direction toward which information is propagated.

5. Restoration

In [13] we proposed our restoration algorithm to restore quasihorizontal scratches. In this paper we present the extended version for scratches with any orientation.

Our method can be divided into two substeps:

(i) direction estimation,

(ii) pixel filling.

5.1. Direction Estimation. This step takes as input the image and the scratch mask. The scratch line divides the neighbor area into two subareas (side 1 and side 2 in Figure 9). Goal of this step is to estimate the direction toward which information is propagated from one side to the other side of the scratch.

For each pixel in the scratch mask, we select two windows (R_1 and R_2), nearby the scratch, one per side. Window size depends on the average gradient of the pixels in the area, as described as follow. A block-matching approach is used to test all the possible candidate vectors that link a block in one side with a block in the other side, starting from the pixel position. Only blocks that are symmetrical with respect to the pixel position are compared, in order to avoid annoying artifacts (i.e., barrel distortion) into the reconstructed area.

The vector that minimizes the Sum of Absolute Differences (SAD) of the pixels in the two end-point blocks is chosen as the most probable direction vector toward which information is propagated:

$$\hat{v} = \arg\min_{\overline{v}\in C} \sum \left| B_1\left(\overline{P}+\overline{v}\right) - B_2\left(\overline{P}-\overline{v}\right) \right|, \qquad B_1 \in R_1 \; B_2 \in R_2, \tag{6}$$

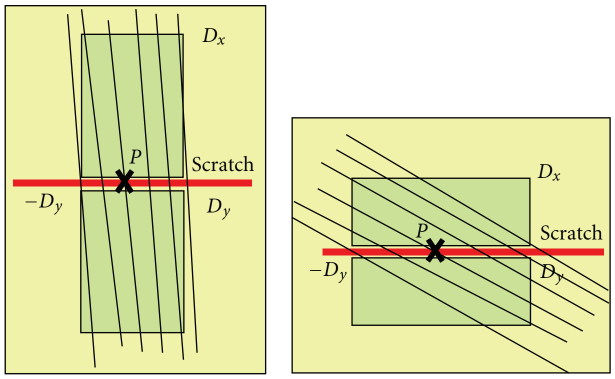


FIGURE 10: Windows are resized to adapt to information inside the area. In case of vertical lines, height is reduced and width increased, and vice versa.

where

(i) B_1 and B_2 are two blocks of pixels symmetrical with respect of P in the scratch, according to the vector v;

(ii) C is the set of possible candidate vectors, with the restriction that B_1 and B_2 must be inside R_1 and R_2, respectively.

For simplicity, we rotate the area nearby the scratch by an angle opposite to the slope of the scratch, and we apply our previous method which was specifically designed for quasi-horizontal scratches. We look for the vector components d_x and d_y:

$$\left(d_x, d_y\right) = \arg\min_{k_x,k_y} \sum_{k_x=0}^{D_x} \sum_{k_y=-D_y}^{D_y} \mathrm{SAD}\left(x, y, k_x, k_y\right),$$
$$\mathrm{SAD}() = \sum_{i=-1}^{1} \sum_{j=-1}^{1} \left| p\left(x - k_x - \frac{w}{2} + i, y - k_y + j\right) - p\left(x + k_x + \frac{w}{2} + i, y + k_y + j\right) \right|, \quad (7)$$

where

(i) w is the scratch width at the pixel position and is updated after each horizontal scan;

(ii) k_x and k_y are the candidate direction vector components;

(iii) D_x and D_y are the vertical and the horizontal sizes of the area into which matching is searched. D_x and D_y depend on the mask width and on the gradient vector as discussed above.

In fact, if horizontal gradient is higher than vertical gradient, there are more vertical than horizontal lines in the area. Therefore we reduce width and increase height of the window, in order to help the reconstruction of vertical lines, and vice versa (Figure 10).

5.2. Pixel Filling. The pixel filling step works in the three RGB color channels. For each color channel, we consider the center pixels of the two blocks detected in the previous step. We assign to the pixel-to-fill the median value between these two points and the average of the two vertically closest pixels outside the scratch mask:

$$\begin{aligned} \mathrm{up} &= p\left(x - d_x - \frac{w}{2}, y - d_y\right), \\ \mathrm{down} &= p\left(x + d_x + \frac{w}{2}, y + d_x\right), \\ v &= \frac{\left(p(x-1, y) + p(x+w, y)\right)}{2}, \\ p^{'(x,y)} &= \mathrm{median}(\mathrm{up}, \mathrm{down}, v). \end{aligned} \quad (8)$$

d_x and d_y are the components of the estimated direction vector. If up and down are similar, the new pixel value will be one of them. If they are very different, we choose the vertical interpolated value, which is probably in the middle of the other two. This solution introduces no artifacts but only few blurring.

The restored area is then rerotated and replaced into the original damaged area. Finally, a median filter is applied to the boundary of the scratch mask, to remove some residual artifacts. Experiments showed that the quality of the results is independent of the rotation angle.

6. Experimental Results

We tested our method onto two different sets of images: 20 images coming from a database of aerial photos, which present scratches more or less horizontally oriented, 20 crops of old photos which have heterogeneous physical causes, and present scratches with any orientation and width. Some visual results are shown in Figure 12.

With respect to the detection method (examples in Figures 11(b), 11(e), and 11(h)), we measured some parameters (Table 1) to evaluate our experimental results within the two testing sets.

(i) CD is the percentage of correctly detected scratches, with respect to the number of the real ones; percentage of not detected (false negatives) is the complementary value.

(ii) FP is the percentage of false positives, that is, the number of detected scratches that are not real, with respect to all the candidate scratches.

(iii) CP (Contour Precision) is a measurement of the accuracy of our labeling process:

$$R = \frac{n(A_{\mathrm{DS}} \cap A_{\mathrm{RS}})}{n(A_{\mathrm{RS}})}, \quad P = \frac{n(A_{\mathrm{DS}} \cap A_{\mathrm{RS}})}{n(A_{\mathrm{DS}})}$$
$$CP = R \cdot P. \quad (9)$$

(iv) R is the recall, the ratio between the number of pixels in the intersection of the detected and the real scratch

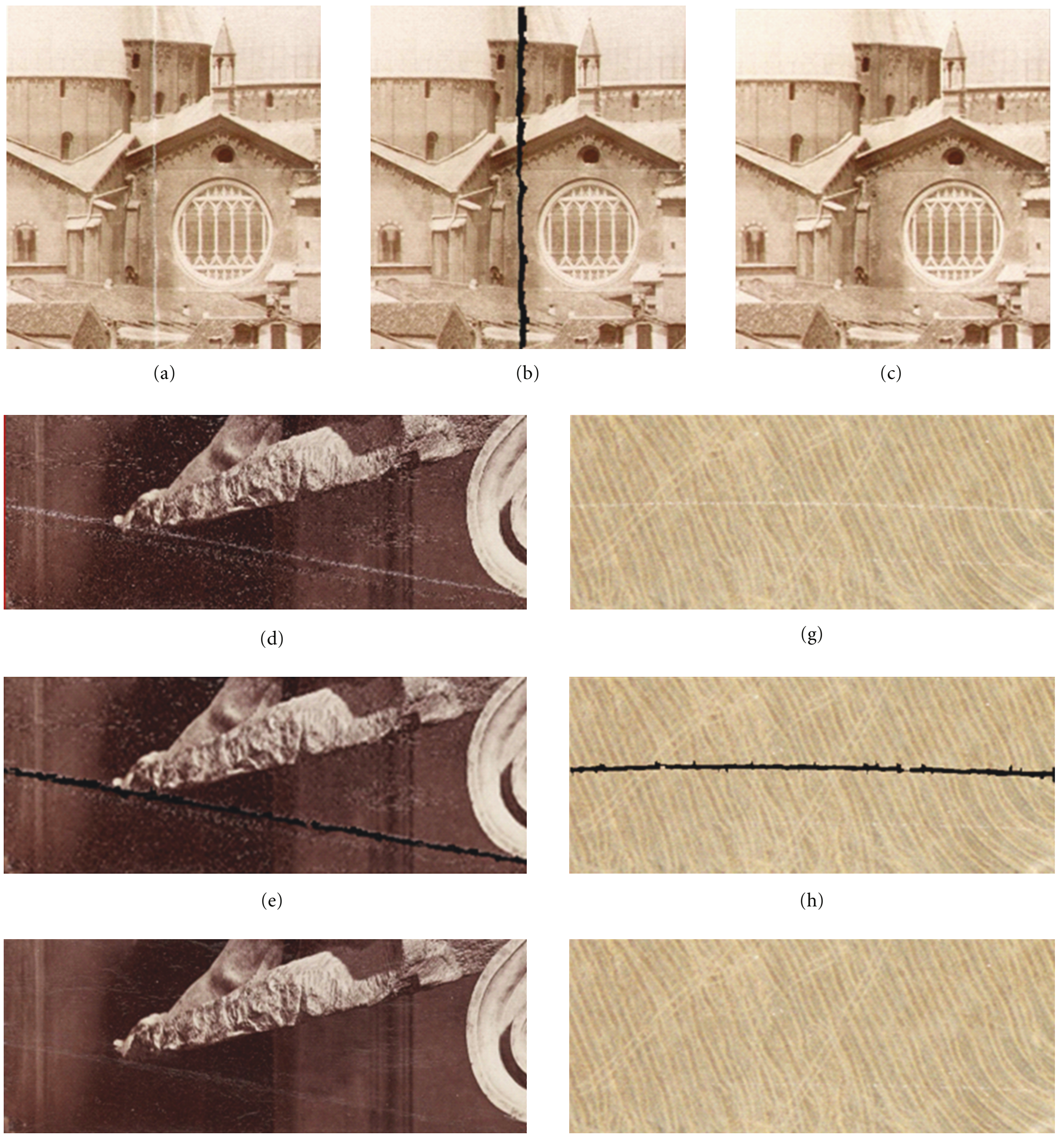


Figure 11: Three examples of scratches ((a), (d), (g)) and the corresponding results: (a) vertical straight line with irregular color and no breaks, from the old photo dataset horizontal, (b) diagonal bright line, with some breaks, (c) curve line with regular color and breaks, from the aerial photo dataset; ((b), (e), (h)) corresponding detected scratches; ((c), (f), (i)) restored images. In figure (f) only the most evident scratch is processed.

and the number of pixels in the real scratch. When it tends to 1, the detected scratch covers the whole real scratch, but it gives no information about pixels outside A_{RS}; if it tends to 0, detected and real scratch have smaller intersection.

(v) P is the precision, that is, the ratio of the number of pixels in the intersection of the detected scratch A_{DS} and the reference scratch A_{RS}, and the number of pixels in A_{DS}. When P tends to 0, the whole detected scratches has no intersection with the real one. If it tends to 1, fewer pixels of A_{DS} are labeled outside AR. Nevertheless this parameter will not assure that the real scratch has been covered.

What can be considered the "real" scratch is a critical issue. For each image in our dataset we manually selected the pixels which belong to a scratch and considered the obtained masks as the "real" contour of the scratch. A manual segmentation process is highly subjective, but it is the only one solution to give a numerical evaluation to our contour drawing method. As expected, we had best

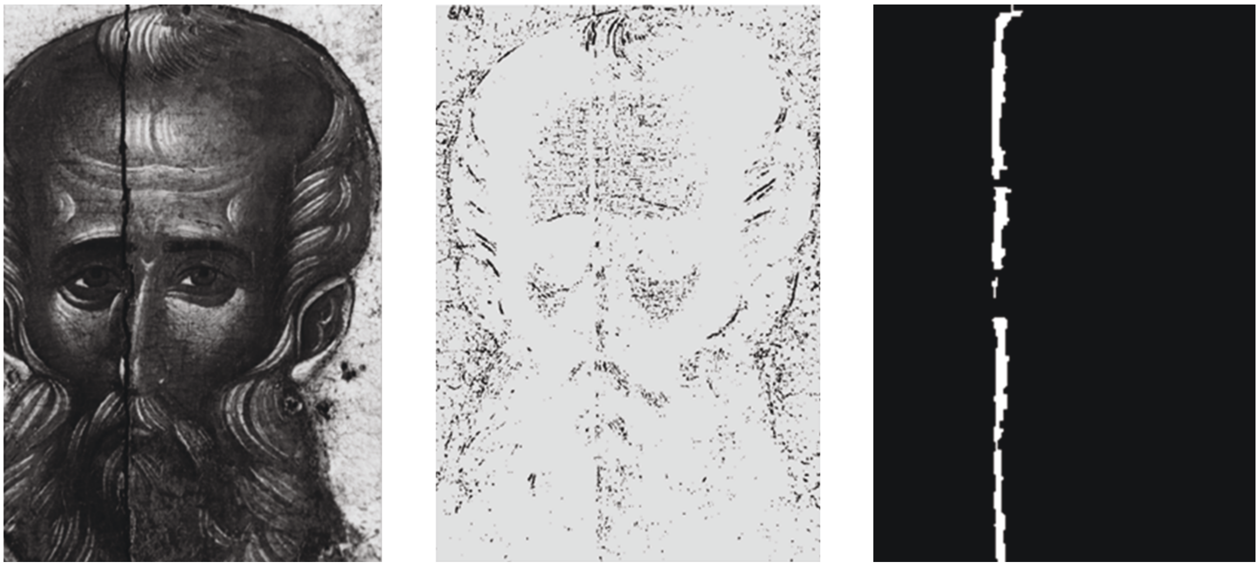

(a) (b) (c)

Figure 12: A damaged painting (a) cracks detected by [5] (b), and "scratch" detected by our method (c).

results for aerial photos since all the scratches in that dataset have more regular features: horizontal direction, white color, and constant width. We noted that many false positives are real lines, which are indistinguishable, for an automatic inspection method, from line scratches.

Table 2 shows the average execution time for each of the steps of our processing method. Note that most of the time is spent in the prefiltering step and in region growing.

Table 1: Quantitative evaluation of experimental results.

dataset\parameters	CD	FP	CP
old photos	81%	28%	81%
aerial photos	85%	17%	91%

6.1. Comparison with Other Works. The choice of a reference method to which we compare our results was a very difficult task. To our knowledge, there are no other approaches which can be applied to our dataset, as none of them deal with the "general" scratch, as we intended. A similar problem is that discussed by Giakoumis et al. [2] which presents a method to process cracks in damaged paintings. Cracks typically result from mechanical/chemical causes as nonuniform contraction in the canvas or wood-panel support of the painting, the evaporation of volatile paint, or external causes, for example, vibrations and impacts. With respect to cracks, which usually have low luminance and thus can be considered as local intensity minima with rather elongated structural characteristics, scratches may have different aspects (Figure 12 shows the differences between these two types of defects in paintings).

Nevertheless, to further evaluate our experiments, we decided to test our detection method with 5 old movie frames and to compare results with those obtained using the approach proposed in [12]. The reference method is chosen as it is one of the most cited between those which use only spatial information.

Within this dataset our results are slightly worse than those of the state-of-the-art approach, as it had been specifically designed for vertical dark scratches in movies (see Figure 13), while our method finds scratches along all the possible directions (see the horizontal line in Figure 13(c)). If we consider only the vertical direction (in the prefiltering step and the Hough-transform), false positives decrease, and results improve and become similar to those of reference approach (Figure 13(d)), spending less execution time (average execution time: 0.6 s versus 1.1 s, testing only vertical direction). No comparison can be made using the CP parameter, because reference method does not deal with the contour finding problem.

7. Conclusions and Future Works

The problem of restoring scratches in still images is much harder to deal with than in old movies, which are typically affected by long vertical thin lines. Scratches in still images can have different, and variable, orientations, thickness, and colors. Furthermore, in still images, no temporal information can be used for the detection. Our method introduces several new contributions with respect to the state of the art. First, our method can detect scratch lines regardless of their orientation. Second, we label pixels that belong to scratch, drawing the defect contour, detecting interruptions, changes in width, and little changes in slope. Moreover tests showed that our method is independent of the scratch color.

The main drawback in the detection step is that it requires user intervention, to select true scratches between a set of candidates, and the starting point. Actually we are

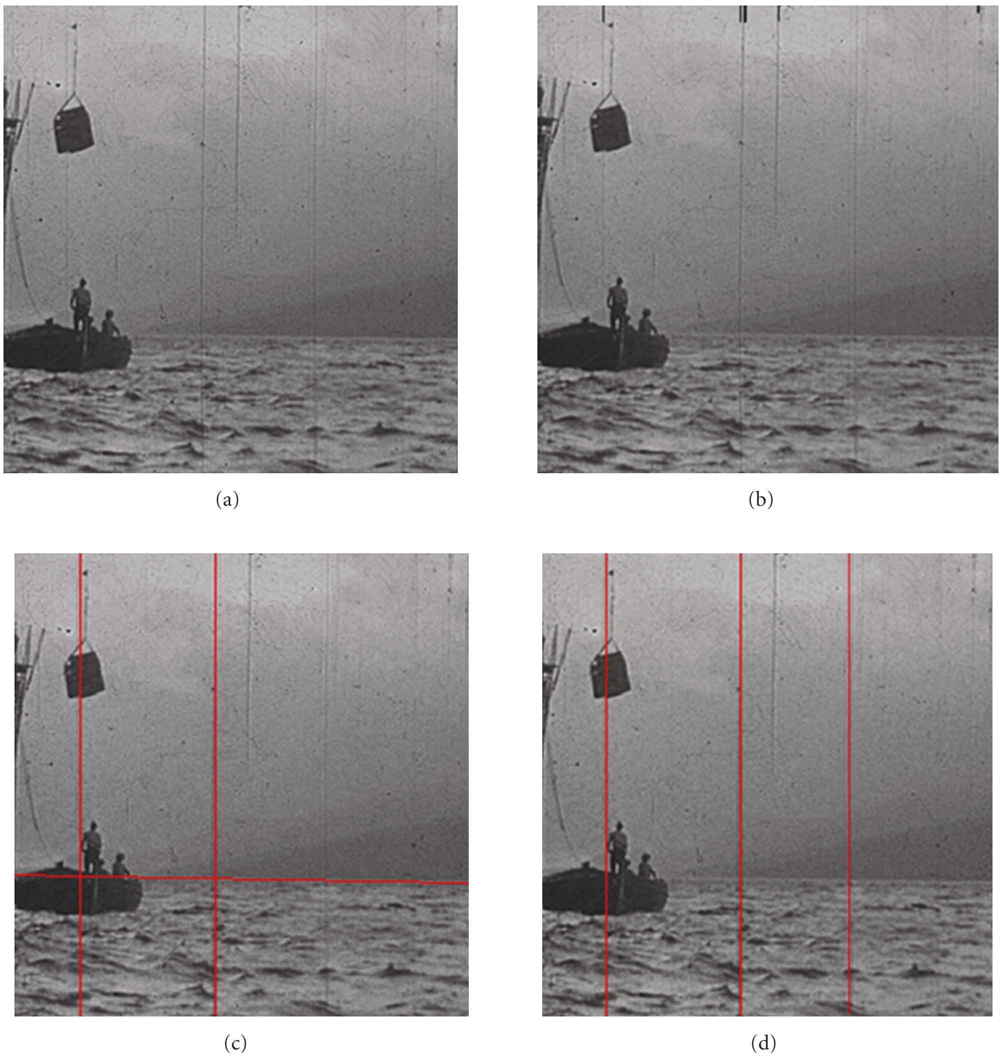


Figure 13: Original old frame with vertical scratches (a). Results applying [5] (b): lines are indicated by a short black line on the top part of the image. Results with our general method (c): it detects also the horizontal line that is not a scratch. Results with our method considering only the vertical direction (d): one more scratch is detected.

Table 2: Average execution time of the steps of our method.

Step	Average execution time (sec)
Prefiltering	6,2
Hough-transform	0,8
Core detection	1,4
Region growing	5,7

working to make our method as more automatic as possible, to select the best starting pixel to draw the contour.

As regards restoration, we proposed a new method, based on block matching. This method ensures the correct reconstruction of lines across the scratch, introducing much less blurring compared to a simple interpolation process. Moreover, our restoration method is very fast, compared with classical inpainting methods, which are typically time consuming. In fact we match only symmetrical blocks, drastically reducing the number of matches to compute. Nevertheless, this method works well only if there a kind of simmetry between the two sides of the scratch, that is, in most cases, a realistic hypothesis, if the scratch thickness is not very large. In case of highly-random textured area, no matching can be found between blocks and vertical interpolation is preferred, and some blurring is introduced. Texture synthesis methods, in these cases, should perform better.

Acknowledgments

The authors wish to thank the Alinari Photo Archives in Florence and SAS S.R.L. in Palermo for having permitted the use of their photos in this research. They also want to acknowledge all the students who collaborate with them for their implementation work.

References

[1] F. S. Abas and K. Martinez, “Craquelure analysis for content-based retrieval,” in *Proceedings of the 14th International Conference on Digital Signal Processing*, Santorini, Greece, July 2002.

[2] I. Giakoumis, N. Nikolaidis, and I. Pitas, “Digital image processing techniques for the detection and removal of cracks in digitized paintings,” *IEEE Transactions on Image Processing*, vol. 15, no. 1, pp. 178–188, 2006.

[3] F. Stanco, L. Tenze, and G. Ramponi, “Virtual restoration of vintage photographic prints affected by foxing and water blotches,” *Journal of Electronic Imaging*, vol. 14, no. 4, 2005.

[4] G. Ramponi and V. Bruni, “Virtual restoration of faded photographic prints,” in *Proceedings of 14th European Signal Processing Conference*, Florence, Italy, September 2006.

[5] E. Ardizzone, A. De Polo, H. Dindo, G. Mazzola, and C. Nanni, “A dual taxonomy for defects in digitized historical photos,” in *Proceedings of the 10th International Conference on Document Analysis and Recognition (ICDAR ’09)*, pp. 1166–1170, July 2009.

[6] L. Joyeux, O. Buisson, B. Besserer, and S. Boukir, “Detection and removal of line scratches in motion picture films,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR ’99)*, pp. 548–553, June 1999.

[7] M. K. Güllü, O. Urhan, and S. Ertürk, “Scratch detection via temporal coherency analysis and removal using edge priority based interpolation,” in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS ’06)*, pp. 4591–4594, May 2006.

[8] L. Tenze and G. Ramponi, “Line scratch removal in vintage film based on an additive/multiplicative model,” in *Proceedings of IEEE/EURASIP Workshop on Nonlinear Signal and Image Processing (NSIP ’03)*, Grado, Italy, June 2003.

[9] A. Kokaram, “Detection and removal of line scratches in degraded motion picture sequences,” in *Proceedings of the 8th European Signal Processing Conference*, vol. 1, pp. 5–8, September 1996.

[10] T. Bretschneider, O. Kao, and P. Bones, “Removal of vertical scratches in digitised historical film sequences using wavelet decomposition,” in *Proceedings of the Image and Vision Computing New Zealand (IVCNZ ’00)*, pp. 38–43, 2000.

[11] V. Bruni and D. Vitulano, “A generalized model for scratch detection,” *IEEE Transactions on Image Processing*, vol. 13, no. 1, pp. 44–50, 2004.

[12] D. Tegolo and F. Isgrò, “Scratch detection and removal from static images using simple statistics and genetic algorithms,” in *Proceedings of the IEEE International Conference on Image Processing (ICIP ’01)*, pp. 265–268, October 2001.

[13] E. Ardizzone, H. Dindo, O. Gambino, and G. Mazzola, “Scratches removal in digitised aerial photos concerning sicilian territory,” in *Proceedings of the 14th International Conference on Systems Signals and Image Processing (IWSSIP ’07)*, pp. 393–396, June 2007.

[14] E. Ardizzone, H. Dindo, G. Mazzola, M. Scriminaci, and M. Vitali, “Multi-directional detection of scratches in digitized images,” in *Proceedings of the 17th European Signal Processing Conference (EUSIPCO ’09)*, pp. 248–252, Glasgow, Scotland, August 2009.

[15] M. Kass and A. Witkin, “Analyzing oriented patterns,” *Computer Vision, Graphics and Image Processing*, vol. 37, no. 3, pp. 362–385, 1987.