# MULTI-DIRECTIONAL DETECTION OF SCRATCHES IN DIGITIZED IMAGES

*E. Ardizzone, H. Dindo, G. Mazzola, M. Scriminaci, M. Vitali*

Dipartimento di Ingegneria Informatica (DINFO) - Università degli Studi di Palermo
Viale delle Scienze, building 6, 90128, Palermo, Italy
phone: +390917028521, fax: + 390916598043,
ardizzon@unipa.it, dindo@csai.unipa.it, mazzola@csai.unipa.it,
mario.scriminaci@gmail.com, monicavit164@gmail.com
web: www.dinfo.unipa.it

## ABSTRACT

*Line scratches are common defects in old archived videos. Many printed images suffer of a similar problem, in most cases due to 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 present a method that consistently modifies and extends our previous proposed algorithm, to detect line scratches in digitized still images. Our method is able to detect scratches regardless of their orientation, and to draw their contour by labeling the pixels belonging to them.*

## 1. INTRODUCTION AND STATE OF THE ART

Scratches are typical damages of old movie films, and look like 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. When digitized, movie films can be processed by digital post-processing techniques in order to reduce or remove these artifacts. To do this, two steps are needed: detection and restoration. In this paper we will present a detection method for scratches in digitized images. The problem of detecting scratches is much harder to deal in still images than in old movies, at least for two reasons: first, scratches manifest in still images with any orientation, color and width (see fig.1); second, no temporal information can be used to process them. Although some authors used a spatio-temporal point of view to process scratches [4], 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 [6] proposed a 2-dimensional autoregressive model for scratch detection and removal, without using information from adjacent frames. Bretschneider et al. [2] proposed a technique based on wavelet decomposition. Bruni et al [3] generalized the Kokaram's model for scratch detection on the hypothesis that a scratch is not purely additive on a given image. Tegolo and Isgrò [7] approach is based on the analysis of the statistics of the image grey levels. All the proposed methods dealt only with long vertical dark (or bright) scratches. The purpose of this paper is to propose a more general approach, which is able to find scratches regardless of their orientation and their color. Furthermore, our method aims to label pixels which are part of the detected line scratch, drawing its contour. To our knowledge, none of proposed methods deepen the analysis of a scratch, but all of them stop detection when only the position of the (vertical) line is found. Our work start from [1], in which we proposed a method to detect quasi-horizontal scratches in digitized aerial photos. This work had been consistently modified and extended in order to process scratches with different aspects.

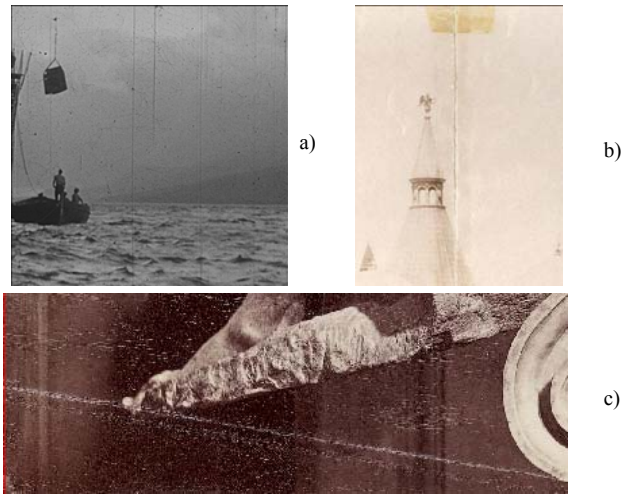



Fig.1 Three examples of images with scratches: a frame of an old movie (a) and two crops from old photos (b and c).

## 2. MULTI-DIRECTIONAL DETECTION

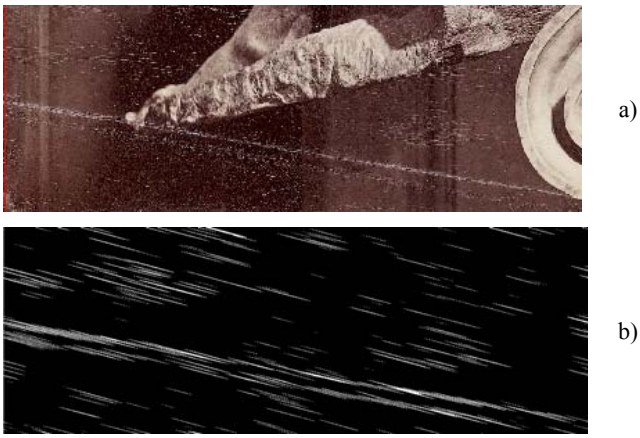
The approach presented in [1] is based on a band-pass filtering, to enhance the horizontal components of the image, and a Hough transformation, to detect candidate line scratches. Therefore, scratches which have slopes with more than 30° are not revealed. This is not enough to achieve our goals. Our extended solution can be divided into three steps: preprocessing, line detection and contour drawing.

### 2.1 Pre-processing

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

a) b) c) d)

Fig.2 Three examples of colored scratches: black (a), grey (b), light blue (c); the output is the same (d)



a)

b)

Fig.3 Input image (a) and one of the output of the pre-filtering step (b) (the direction of the diagonal scratch shifted by 90°).

image its local-mean filtered version), thus eliminating the DC component. Tests showed that this solution makes our approach independent from the color of the scratch (see fig. 2), and helps the next steps in detecting its direction.

Then a bank of oriented band-pass filters is applied. Kass and Witkin [5] affirmed that a line is 90° shifted in the frequency domain and suggest the use of a band-pass filter:

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

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

$$\begin{cases} tx = center\cdot\cos(\theta) \\ ty = center\cdot\sin(\theta) \end{cases}; \qquad \begin{cases} u^* = \cos(\theta)\cdot(u+tx)+\sin(\theta)(u+ty) \\ v^* = -\sin(\theta)\cdot(u+tx)+\cos(\theta)(u+ty) \end{cases} \quad (2)$$

so that the center of the sub-band has $t_x$, $t_y$ coordinates and it is rotated with the same angle. The sub-bands 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 sub-band, so that we chose n=4 to concentrate the filtering in a precise zone of the spectrum. This explains the use of this band-pass filter instead of a Gabor one.

We selected 12 not overlapping filters, to analyze 12 different directions, rotated with 15° each other.

A homomorphic filter is then applied to enhance the 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. The result after the whole filtering process can be seen in fig 3.b.

### 2.2 Line direction detection

After the preprocessing step, we apply the Hough transform to the 12 output binary images, in order to extract any 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 the respect of the input image). Therefore for each direction only this information is analyzed, and saved as a column of 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 (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, and all the values in a neighborhood, are set to zero. This solution is applied to avoid the detection of multiple overlapping scratches (fig.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 threshold, the detection step ends (see fig.5).

### 2.3 Contour drawing

At this point our method requires 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 can be caused by heterogeneous reasons, 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

a)

b)

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

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 sub-steps: 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.

*2.3.1 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°,45°]. 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 search for the 5 nearest pixels (see fig. 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, and a "local" correlation factor between the candidate and the last included pixel:

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

- $s_c(p_i,k)$ is the similarity function for the core detection, evaluated in the candidate point $p_i$ of the column k ; $w_1$ and $w_2$ are two constants, set by experiments, with $w_1 > w_2$ ;
- $c_g$ 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;
- $c_l$ 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;
- $g(p_i, d_s)$ is a Gaussian function, evaluated in $p_i$, with a peak in correspondence to the detected scratch direction $d_s$ and a constant standard deviation.

For each column we select the point, between the 5 candidates, with the maximum similarity value. If this value is lower than a threshold, no pixels are included into the core, but our method continue searching for other points along the direction $d_s$, until the end of the image.

a) b)

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

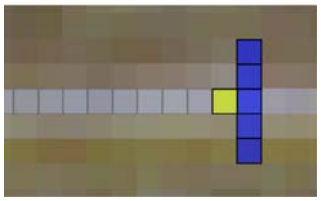


Fig.6 The five nearest pixels (in blue) to the reference point (in yellow), rightward. The same procedure is repeated leftward.

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.

*2.3.2 Region growing*

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

The similarity function is alike eq.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:

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

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

- $s_t(p_i,k)$ is the similarity function for the thickness detection, evaluated in $p_j$ in the column k ;
- $w_1'$ and $w_2'$ are two constants with $w_1' >> w_2'$, set by experiments;
- $c_g$ , $c_l$ , $w_1$ and $w_2$ the same of eq.3;
- $p_{j+1}$ ($p_{j-1}$) is the next upper (lower) pixel to be examined in the same column of $p_j$ ;
- $g(p_j, \mathrm{var}^k)$ is a Gaussian function, evaluated in $p_j$, with a peak in correspondence to the pixel in column k of the scratch core, and a variance which is shown in eq.5 ;
- $\mathrm{var}^k$ and $\mathrm{var}^{k+1}$ (or $\mathrm{var}^{k-1}$) 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);
- $k_{min}$ is the minimum admitted value for the variance.

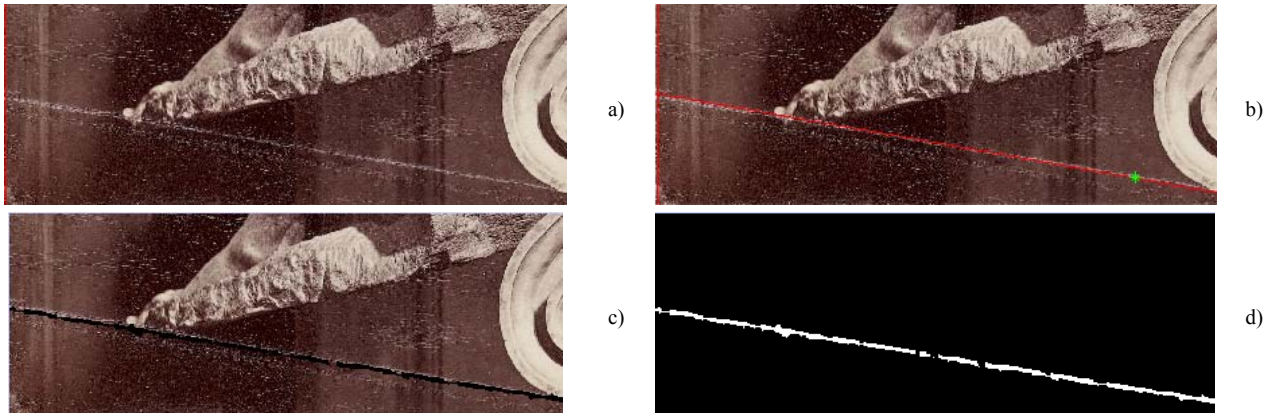


Fig. 7. A complete detection process: the damaged image (a), the detected line (b) and the starting point (the green star on the red line), the detected scratch with its contour (c) and the output binary image (d). Note that the detected mask is interrupted in correspondence to the scratch gaps in the input image.

Our method, for each column, stops when the similarity value of the next candidate pixel (upward or downward) is lower than a threshold. 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 is to create a binary image showing the result of this labeling process. Fig. 7 shows an example of a complete scratch detection process. Note that (fig.7.d) the detected mask breaks in correspondence to the scratch interruptions in the input image.

## 3. 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 fig. 8. Table 1 shows some parameters we measured to evaluate our experimental results within the two testing sets:

- 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;
- 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;
- CP (Contour precision) is a measurement of the accuracy of our labeling process:

$$In = \frac{n(A_{DS} \cap A_{RS})}{n(A_{RS})}; Out = \frac{n(A_{DS} \cap \overline{A}_{RS})}{n(A_{DS})} \qquad 6)$$

$$CP = In \cdot (1 - Out)$$

- *In* is the ratio between the number of pixels in the intersection of the detected and the real scratch and the number of pixels in the real scratch. When this parameter tends to 1, the detected scratch covers the whole real scratch, but nothing can be said about pixels outside $A_{RS}$; if it tends to 0 detected and real scratch have smaller intersection;

Table 1. Quantitative evaluation of experimental results

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

Table 2. Avg execution time of the steps of our method

| step | avg exec time (sec) |
|---|---|
| Pre-filtering | 6,2 |
| Hough-transform | 0,8 |
| Core detection | 1,4 |
| Region growing | 5,7 |

- *Out* is the ratio between the number of pixels of the detected scratch which are not in the real scratch, and the number of pixels in the detected scratch. When this parameter tends to 1, the whole detected scratch has no intersection with the real scratch. If it tends to 0, fewer pixels are labeled outside the real scratch area. Nevertheless this parameter will not assure that the whole reference 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 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 step our processing method. Note that most of the time is spent in the pre-filtering step and in region growing.

Moreover, we tested our method with 5 old movie frames and compared our results with those obtained using the algorithm proposed in [7], which had been specifically designed for vertical dark scratches in movie (see fig.9). Within this dataset our results are slightly worst than that of the state of the art approach, because scratches can be found along all the possible directions (see the horizontal line in

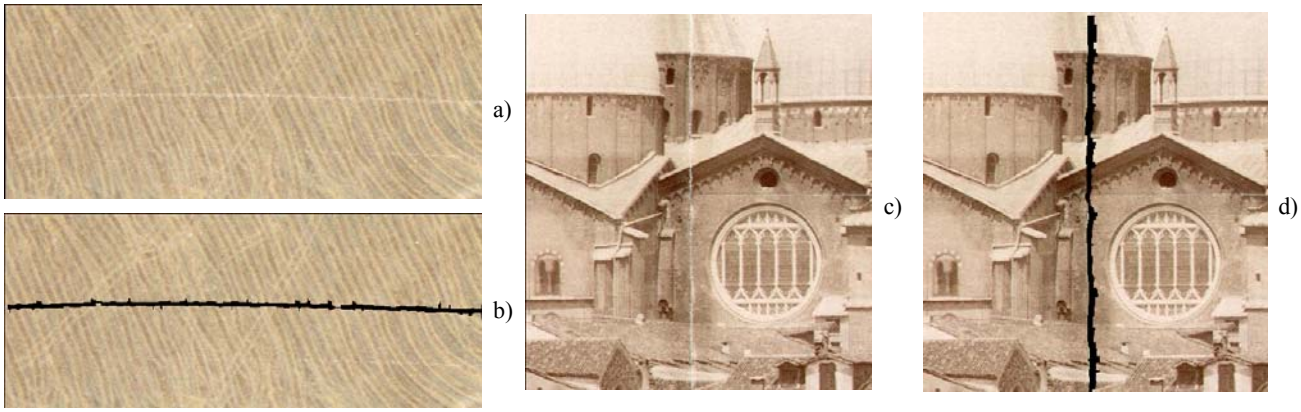



Fig. 8. Two examples of scratches (a,c) and the corresponding results: (a) horizontal, curve line with regular color and breaks, from the aerial photo dataset; (b) vertical straight line with irregular color and no breaks, from the old photo dataset

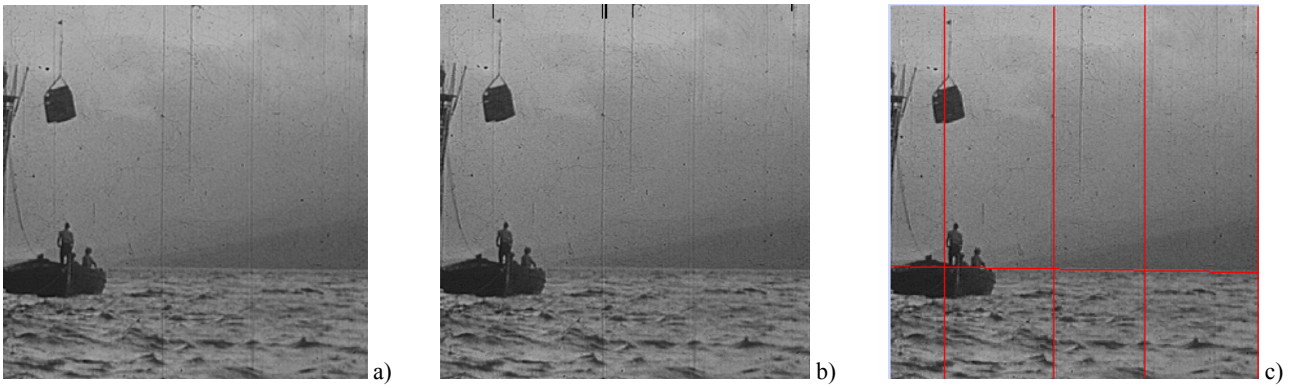



Fig. 9. Our method applied to an old movie frame (c), and compared to the results obtained using a state of the art approach (b). Detected scratches in b are highlighted by a short black line in the top part of the image.

fig. 9.c). If we consider only the vertical direction (for the pre-filtering step and the Hough-transform), false positives disappear, results improve and become similar to those of other approach, with less execution time (average execution time: 0.6 s vs. 1.1 s, testing only vertical direction). No comparison can be made using the CP parameter, because the state of the art method does not deal with the contour finding problem.

## 4. CONCLUSIONS AND FUTURE WORKS

The problem of detecting scratches in still images is much harder to deal than in old movies, which typically are 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. Two are the new main contributions of our approach 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 from the scratch color. To make our method as more automatic as possible, the choice of the starting pixel for the contour drawing step is the most critical point. Actually we are working on a solution for automatically suggesting to the user a starting point, and to develop a specific restoration algorithm, to create a complete methodology for scratch processing.

## ACKNOWLEDGEMENTS

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.

## REFERENCES

[1] E. Ardizzone, H.Dindo, O.Gambino, G.Mazzola "Scratches Removal in Digitised Aerial Photos Concerning Sicilian Territory" in *Proc. IWSSIP 2007*, pp. 393-396

[2] T. Bretschneider, O. Kao, P. Bones "Removal of vertical scratches in digitised historical film sequences using wavelet decomposition", in *Proc. IVCNZ 2000*, pp. 38-43

[3] V. Bruni, D. Vitulano, "A Generalized Model for Scratch Detection", *IEEE Transactions on Image Processing*, Vol. 13, No. 1, January 2004.

[4] L, Joyeux, O. Buisson, B. Besserer, S. Boukir, " Detection and removal of line scratches in motion picture films", *CVPR'99* , pp. 548--553.

[5] M. Kass and A. Witkin "Analysing oriented patterns Computer Vision", *Graphics and Image Processing*, 37 (1987) 362-397

[6] A. Kokaram. "Detection and removal of line scratches in degraded motion picture sequences", in *Signal Processing VIII*, volume I, pages 5-8, September 1996.

[7] D. Tegolo, F. Isgrò, "Scratch detection and removal from static images using simple statistics and genetic algorithms", in *Proc., ICIP 2001*, pp. 265-268.